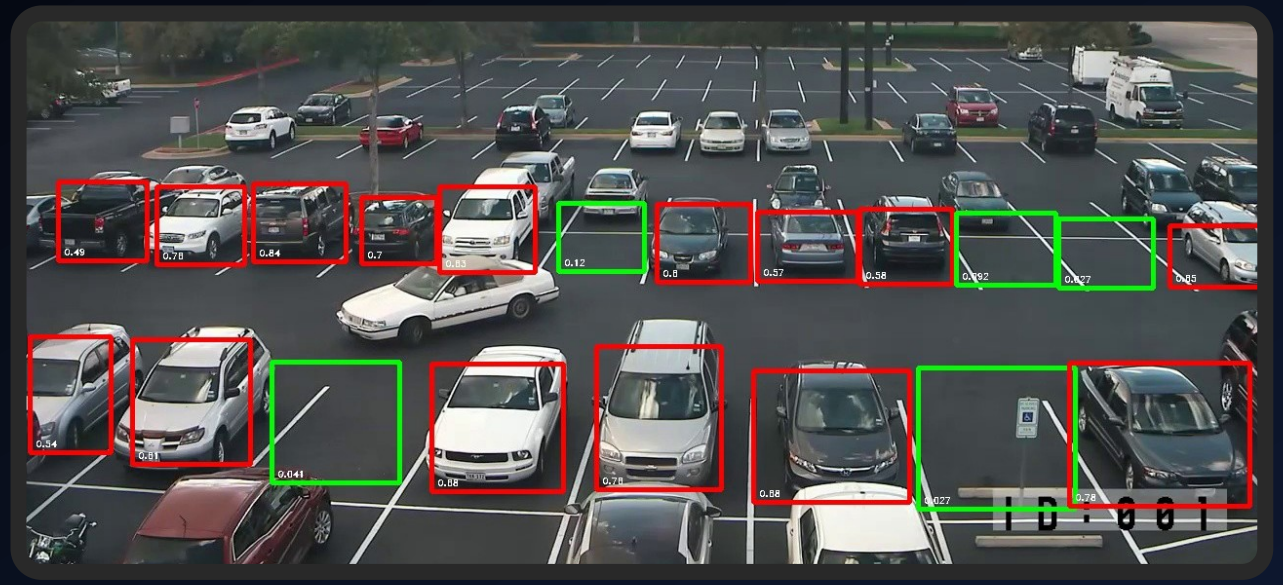
Parking Space Detection

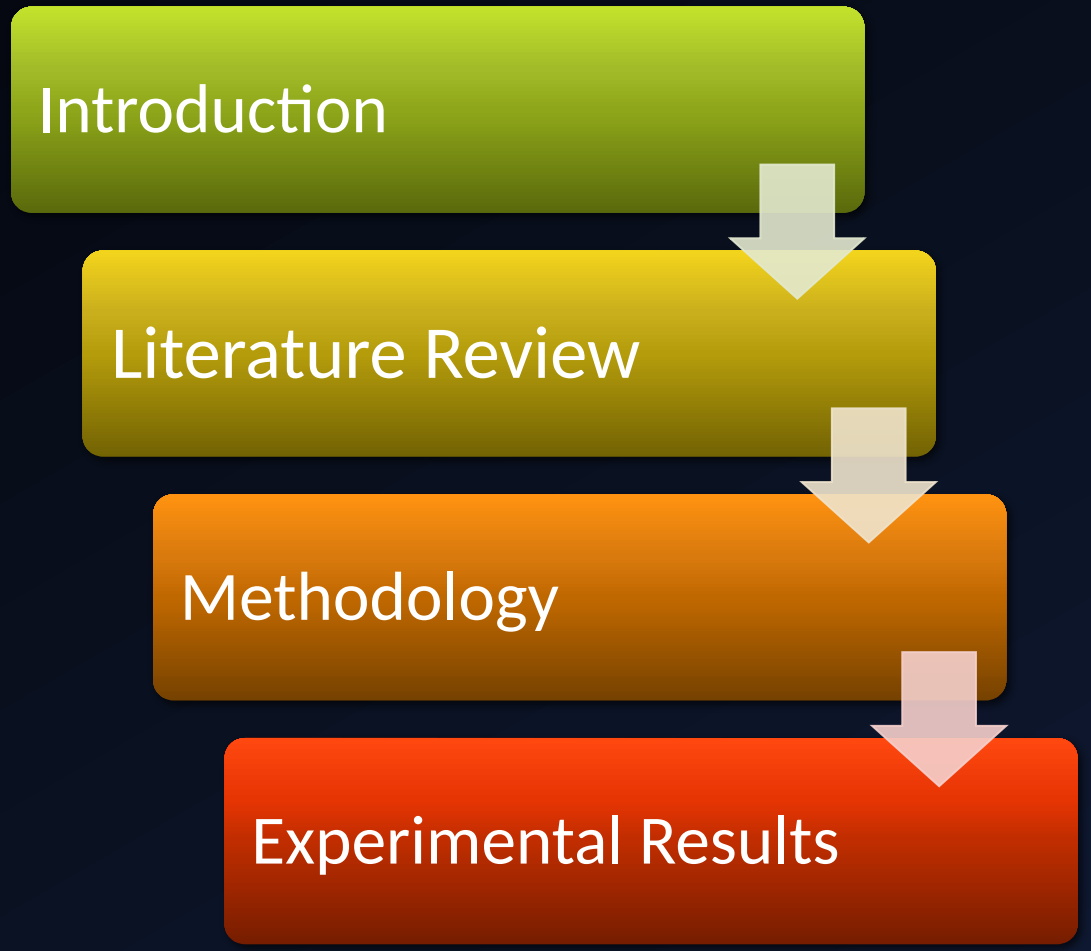
SUPERVISOR
DR/ GAMAL THARWAT



Project Team

1. MAHMOUD OSAMA MOHAMED
2. MOHAMED MAGHAWRY ISMAIL
3. MAHMOUD ABDULRAHMAN MAHMOUD
4. ISLAM MOHAMED MOHAMED SOLIMAN
5. MAHMOUD REZK SAAD

CONTENTS


Introduction
Literature Review
Methodology
Experimental Results

ABSTRACT

- Finding empty parking slot in less time with fewer efforts is a difficult task for drivers mostly in metropolitan areas. Traffic congestion may occur due to unavailability of empty parking slots. This paper represents an approach for smart parking system which facilitates drivers with available empty parking slots. The system is based on image processing. Using video surveillance, it captures images and using detection it notifies the vehicle driver whether the parking lot is available or not. This system will make parking of vehicle more efficient and less complex by providing a better solution to the existing system
- We use CNN and OpenCV Library In Python

Introduction

Overview

- The main aim of this project is to reduce the traffic in the parking place. Normally we can see in the multiplexes, cinema halls, large industries, and function halls there is problem they have to go and search which line is empty and which line having place to park the vehicle, for parking then they need workers for parking in correct position it is the money consumed process. So to avoid this problem Car Parking System project is implemented.

Overview

- Sometimes, there may be an empty space in the parking lot but the driver who sits in the car cannot know where the empty space is exactly located. The reason behind that may be the free spot is far away from the car, or may be the parking space is not visible because of some other cars or big objects hiding the empty space. In the previous days, or may be at some parking lots now there used to be a person who manages the parking spaces, in fact the person who manages also do not have the information about where the empty space is located, they get to know only the total number of empty spaces so the drivers themselves have to look around parking lot and search for the empty space, meanwhile other driver arrives there are many losses such as time, fuel and also temper. So, for these kinds of issues the researchers have developed some solutions like parking lot recognition techniques by making use of some objects like video cameras and sensors which are used to detect an empty and occupied state of the parking space.

Problem Statement

- As the population grows there is an increase in the private vehicles which caused many issues to the people at the crowded areas such as shopping malls, restaurants and universities parking lots. This results in time waste for the drivers in search for the empty parking space which includes atmospheric pollution and frustration of the drivers. On the other hand, most of the other systems show only the total number of spaces empty or occupied rather than showing the exact location.

Objectives

- • Design a model which can efficiently detect the Empty and Occupied status of the parking space in an image and a video by use of YOLOv3 object detection.
- • Detect a car parking lot occupancy in an image and in the video whether it is Empty or Occupied by using YOLOv3 detector which is represented by overlapping the output in OpenCV and to collect data which is good and large enough to train the model.

Limitations

- In the YOLO algorithm, the model predicts and verifies the overall image and video in relation to the training data it verifies in the camera point of view at the time of the test.
- Unless a few images of the test environment are included in the training dataset, the YOLO model cannot get a good detection.
- Resolution values may change depending on lighting conditions.

Scope

- In future the model can be trained by increasing the images in the dataset and making huge dataset in such a way that it can detected at different locations and with different environment.
- The other scope is that the system can be developed in such a way that it could give directions to the drivers to the nearby available empty space by using deep learning and AI methods.

why we use CNN ?

- Convolutional Neural Network.
- This Make Greeting Results More Than Other Technique of CPD(Car Parking Detection)
- this technique work by solve the issues will face the CPD like :-
- parking displacement
- non-unified car sizes
- interobject occlusion
- in this project we will perform a Realtime CPD based on CNN by taking videos from surveillance cameras and detecting the occupancy of park by checking the parking space is empty or occupied.
- we will training the park with YOLOV3 model

Literature Review

Background

- solutions for the parking lot problems are categorized into 3 groups:-

- Counter-based
- Sensor-based
- Vision-based

1- Counter-based

- Counter-based parking space is a system that uses a counter to keep track of the number of available parking spaces in a given area. The counter can be either electronic or manual, and it is typically placed at the entrance of the parking area.
- When a car enters the parking area, the counter is updated to reflect the reduction in available parking spaces. Similarly, when a car exits the parking area, the counter is updated to reflect the increase in available parking spaces. This information is then displayed to drivers looking for a parking space, allowing them to quickly determine whether there are any available spots.

- Counter-based parking space systems are typically used in areas where there are a large number of parking spaces, such as parking garages, parking lots, and public parking areas.
- They are also commonly used in cities to manage on-street parking spaces.
- One of the advantages of counter-based parking space systems is that they can help reduce traffic congestion by allowing drivers to quickly find available parking spaces, rather than circling around looking for a spot. Additionally, they can help reduce air pollution and greenhouse gas emissions by reducing the amount of time that drivers spend looking for parking.

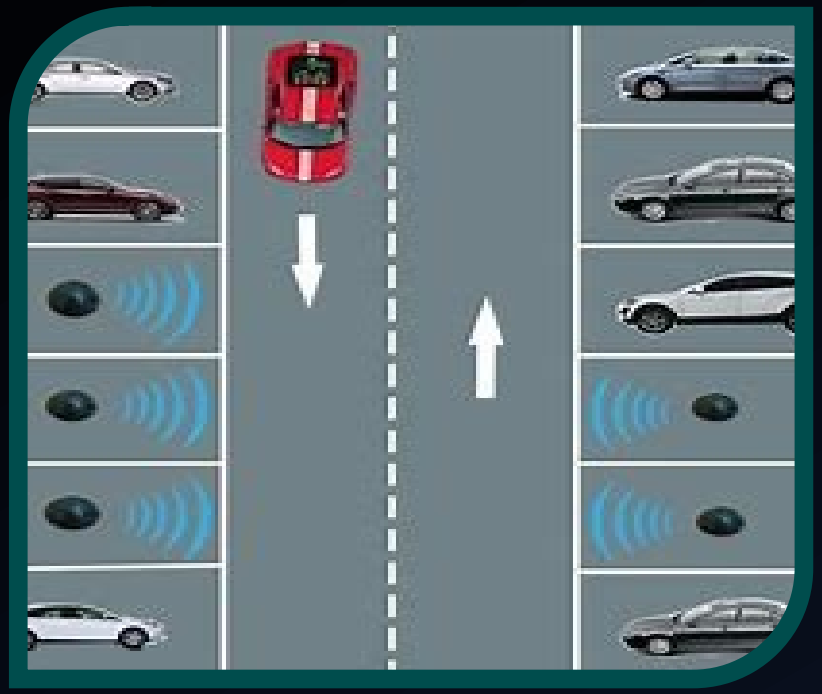
- Overall, counter-based parking space systems can help make parking more efficient and convenient for drivers, while also benefiting the environment and reducing traffic congestion.

- Disadvantage: we can't use this system on streets and residential parking spaces.

Occ Count: 0 Vac Count: 0
Detection
Threshold

2- Sensor-based

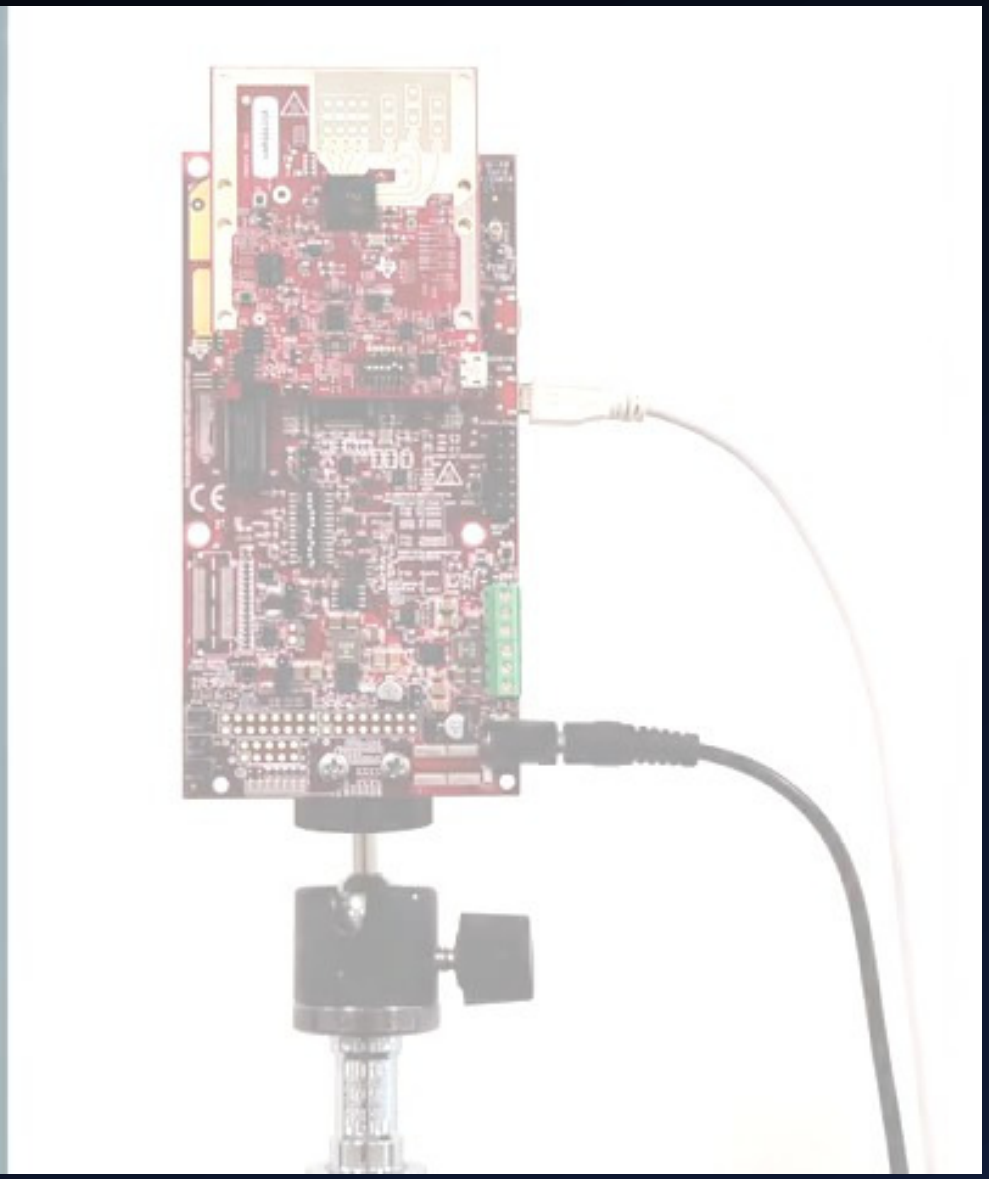
- In the sensor-based systems there are wire and wireless based sensors in which wired systems which mainly depend ultrasonic, infrared light and the wireless based on the magnetic-based sensors which are fixed at the parking lot. Both the wired and wireless are being used mainly at the parking lot like shopping malls which are indoor. Since, these are mainly used at the indoor environments these ground sensors can be used to decide the position of the parking spaces.
- So, the solution for this is to use application-focused sensors which can be like magnetic or infrared sensors and pneumatic road tubes. So, these kinds of sensors are very rarely used in some cities. Some paperwork will be required for installing these sensors for the parking space detection and time will also be wasted and the drivers will face some issues while they are willing to park their cars when the installation of the sensors are going on.



Advantages	Disadvantage's
There is high parking efficiency.	It may be a bit confusing for unfamiliar users.
There is no need for driving while looking for an available space.	It requires high installation and maintenance cost.
There are less chances for vehicle vandalism.	It is not recommended for high peak hour volume facilities.
There is a minimal staff requirement if it is used by known parkers.	

Parking garage occupancy sensor using TI mmWave radar sensors

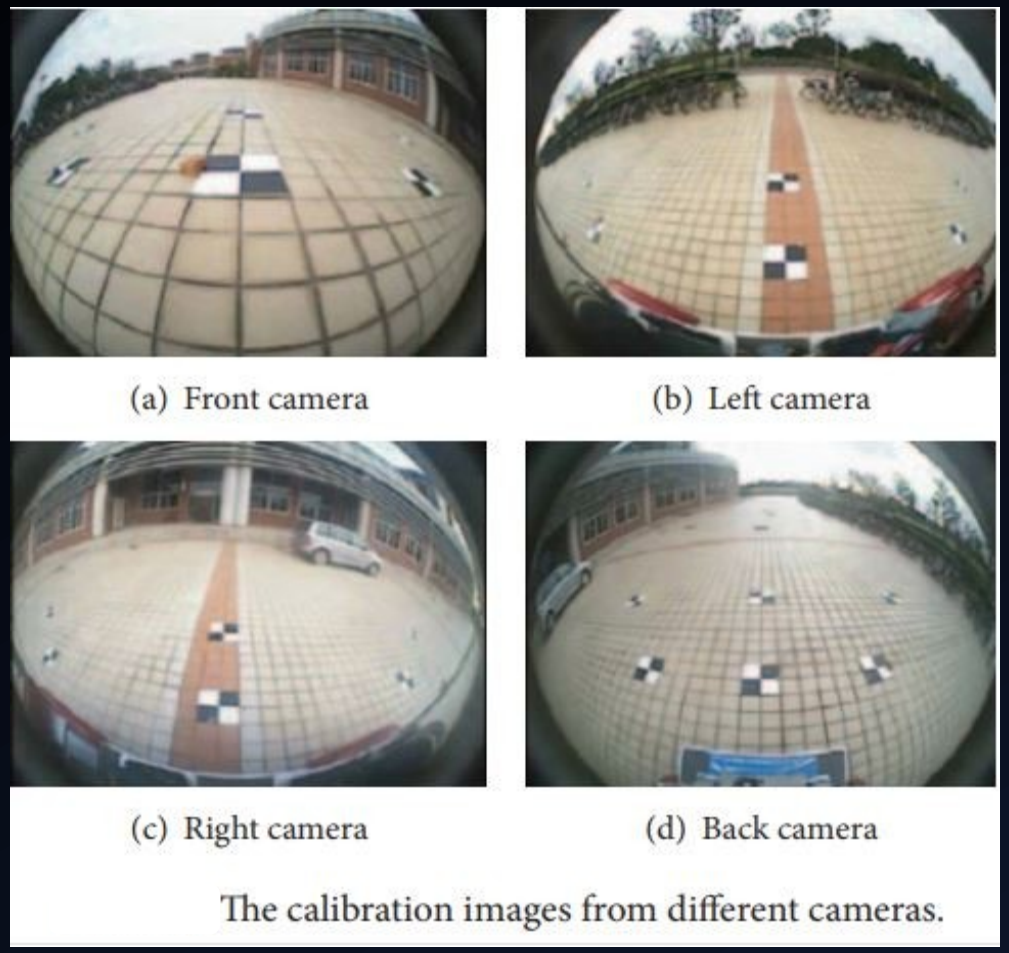
- Low-cost, single sensor to detect occupancy of up to 4 parking spaces
- Technology to sense if a vehicle is blocking a parking space
- Immune to environmental changes like light, smoke etc
- Non-invasive to protect privacy



- **Related Work**
- Author: Sangwon Lee, Dukhee Yoon, Amitabha Ghosh
- in this paper, they used the magnetic sensors and the ultrasonic sensors for better accuracy and quality recognition of the cars in the parking space.
- Author: Yang and Kim (2020)
- proposed a system that uses a combination of ultrasonic sensors and a stereo camera to detect parking spaces and obstacles around the vehicle. The system then generates a parking trajectory and controls the vehicle to park in the designated space.

Parking Based on a Bird’s Eye View:

- Why this method is better than Vision sensors, such as cameras :
- in spite of low cost as well as long life and their precision is also fairly high. In addition, they can provide real-time vision assists and rich image information to drivers.
- But vision sensors have poor performance in darkness conditions which have no additional light sources and it is very difficult to maneuver vehicles by using one single camera, since the blind spots are inevitable in some complicated conditions, such as narrow valleys and the reverse parking. So a low-cost bird’s eye view vision system is constructed by installing four fisheye cameras

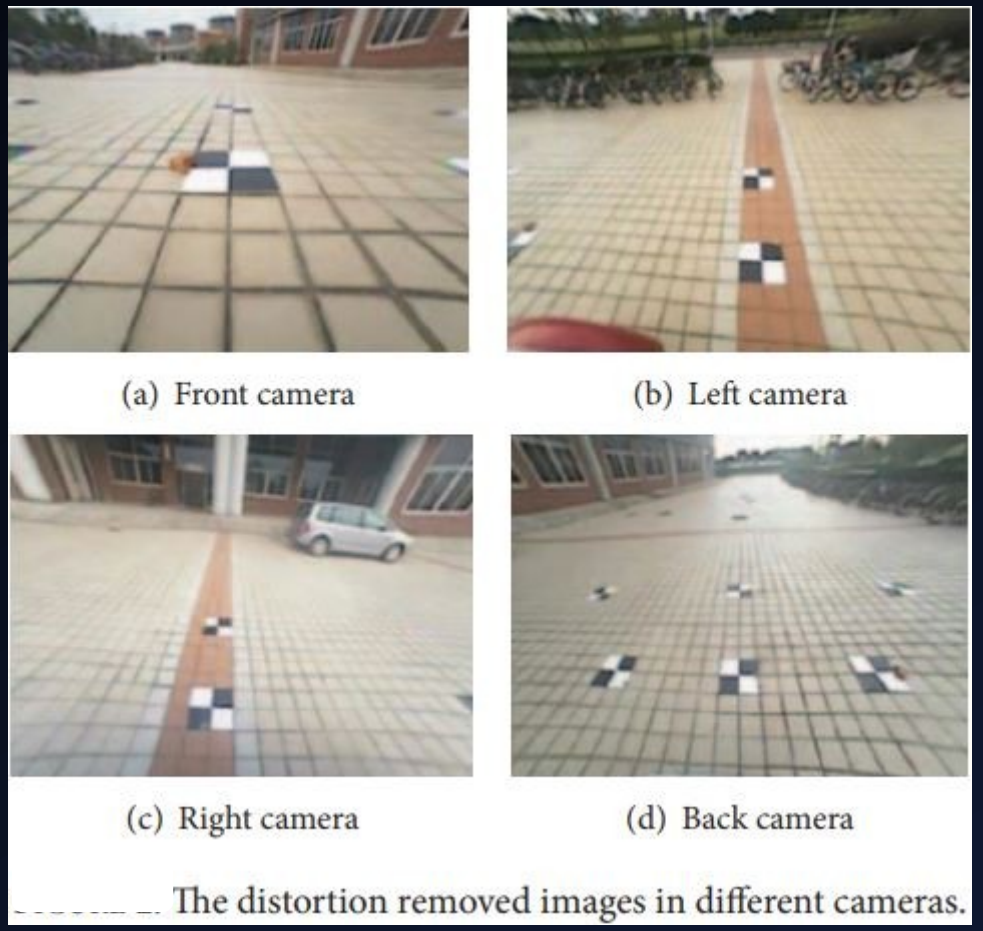


The calibration images from different cameras.

- The Bird’s eye view vision used in the proposed system is based on four fisheye cameras with 180-degree field of view around vehicle. The system consists of two phases:
- calibration and image mosaicking.

• Calibration.

The camera imaging model is geometric mapping from three-dimensional space to two-dimensional pixel space. Camera calibration is to find out the parameters of this mapping relation, that is, camera parameters, which can be divided into intrinsic parameters and extrinsic parameters. The original images captured from the fisheye cameras are distorted after calibration.



The distortion removed images in different cameras.

- **Image Mosaicking**

Image mosaicking is an effective means of constructing a single seamless image by aligning multiple partially overlapped images.
based on the Levenberg-Marquardt algorithm is used to combine four individual images from fisheye cameras into one omnidirectional bird's eye view image.

Bird's eye view vision system. Images F_f, F_l, F_r, F_b are captured from the front, left, right, and back fisheye cameras mounted on the CyberC3 vehicle. The image in the center is the bird's eye view image synthesized from the four fisheye images.

- # Parking Space Detection

1.Overview Generally the parking space is often located on one side of the moving vehicle. Therefore,
in order to improve the real-time performance of the automatic parking, only the fish eye cameras on the parking space side will be used to detect the free parking space,
while all of the cameras are used when the parking spaces are around the car.
Subsequently, **the IPM (inverse perspective mapping)** images are used as the input for the free parking space detection

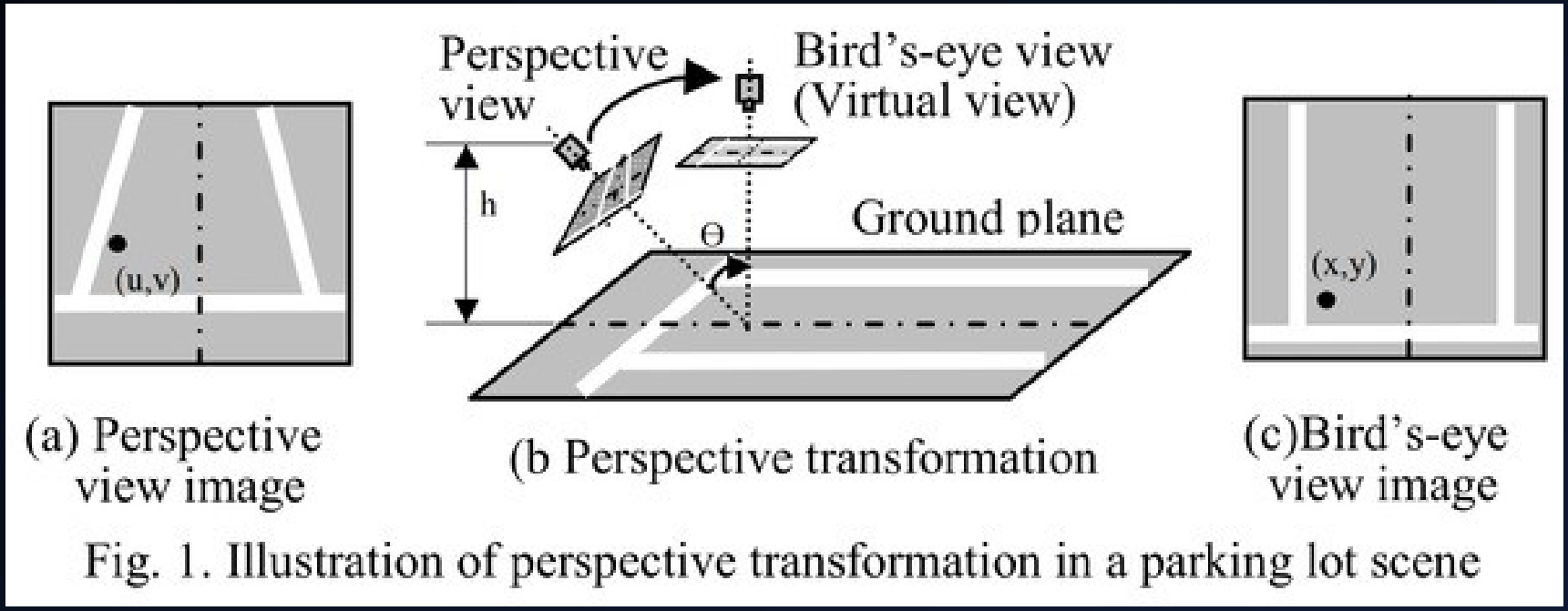
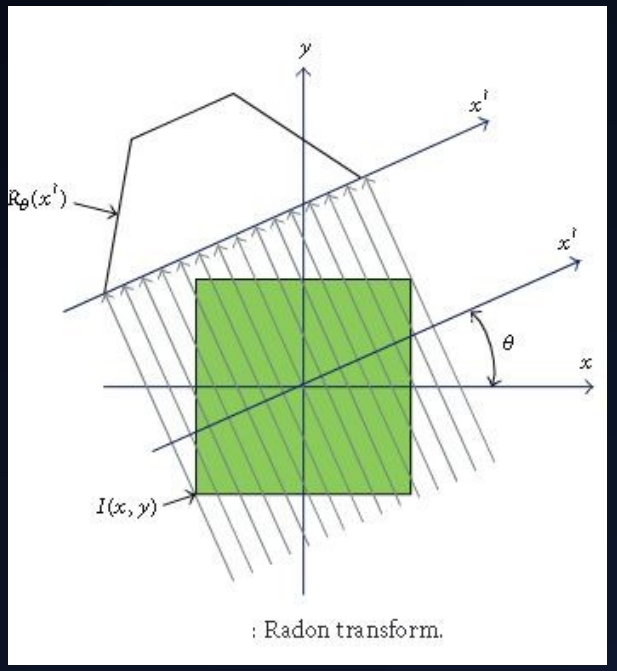


Fig. 1. Illustration of perspective transformation in a parking lot scene

2.Radon Transform:

Applying the Radon transform in an image $I(x,y)$for a given set of angles can be regarded as computing the projection of the image along the given angles. The resultant projection $R(\square,\square)$ is a line integral which is the sum of the intensities of the pixels in each direction. In other words, the liner integral value$R(\square,\square)$ is the projection of the geometry image along the direction $\square$



: Radon transform.

The formulation of the Radon transform is as follows→

$$\rho = x \cdot \cos\theta + y \cdot \sin\theta,$$

$$R(\theta,\rho) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} I(x,y) \cdot \delta(\rho - x \cdot \cos\theta - y \cdot \sin\theta)\,dx\,dy,$$

Where $\square$ is the Dirac Delta function.$(\square,\square)$ is the parameter space about the Radon space.$\square(\square,\square)$ is the value of Radon space at $(\square,\square)$ points.

3.Features Extraction:

Vehicle parking space markings are often constructed with several fixed widths and colored parallel line segment pairs. According to the Radon transform, the lines are mapped to the bright spots (BS) in the Radon space. As shown in Figure 1, they are in one-to-one correspondence. It should be noted that the parallel line segment pairs in the edge image will produce BS pairs in the Radon space. The two BS points in one pair have the same value □ and a fixed width. Based on this feature, a method to detect the center of the BS pair in the Radon space is proposed. Considering the influence of noises, clutters, and vehicles in the edge image, it is necessary to take the area around the BS as the region of interest. The detector structure is shown in Figure 2. HBS is the high bright spot of the pair and LBS is the low bright spot of the pair. PD is the two BS pixel distances of the fixed width mapped from the line pair

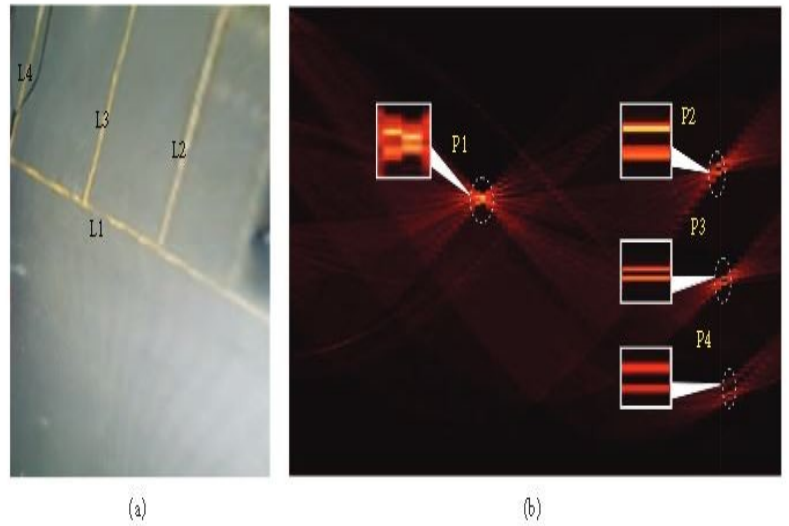


figure1 a) Lines L1, L2, L3, and L4 are the parking space marking lines in inverse perspective image. (b) P1, P2, P3, and P4 are the bri
pot pairs of the lines corresponding in Radon space.

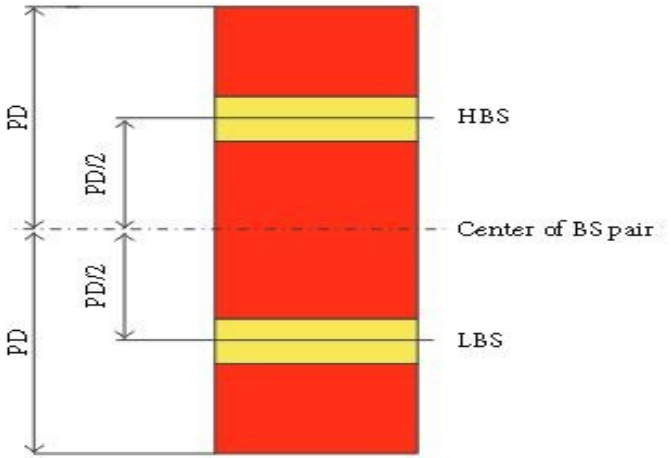


figure2 10: The structure for BS pairs detection. The yellow color i
the BS pair in Radon space.

With known parking space orientation, the set C can be obtained by series candidate feature points about the possible parking space line segment pairs. The detection result is shown in Figure 3.

It should be noted that a number of factors, such as the car body lines, shadows, and the wear of the line markings, may often affect the detection accuracy, as shown in Figure 4. In order to get accurate parking space marking lines, some measures to remove the noise factors should be taken.

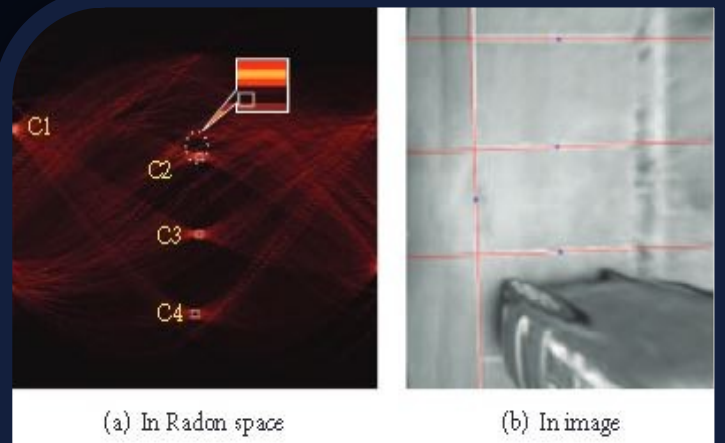


figure3 The center detection result of the BS pairs of parking space lines. (a) C1, C2, C3, and C4 are the candidate points detected by the proposed method in Radon space. (b) The red lines are the detection result of the points in image space.

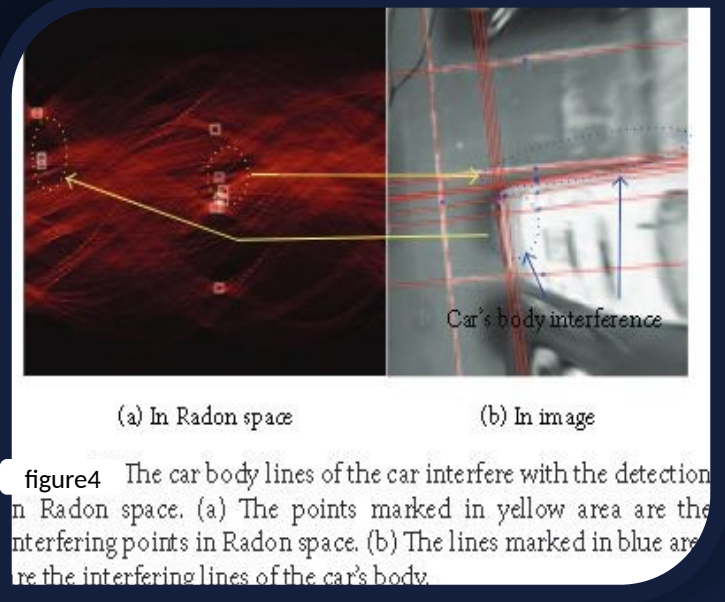


figure4 The car body lines of the car interfere with the detection in Radon space. (a) The points marked in yellow area are the interfering points in Radon space. (b) The lines marked in blue are the interfering lines of the car's body.

Figure 5 shows the proposed methods take good performance in different scenes for parking space detection

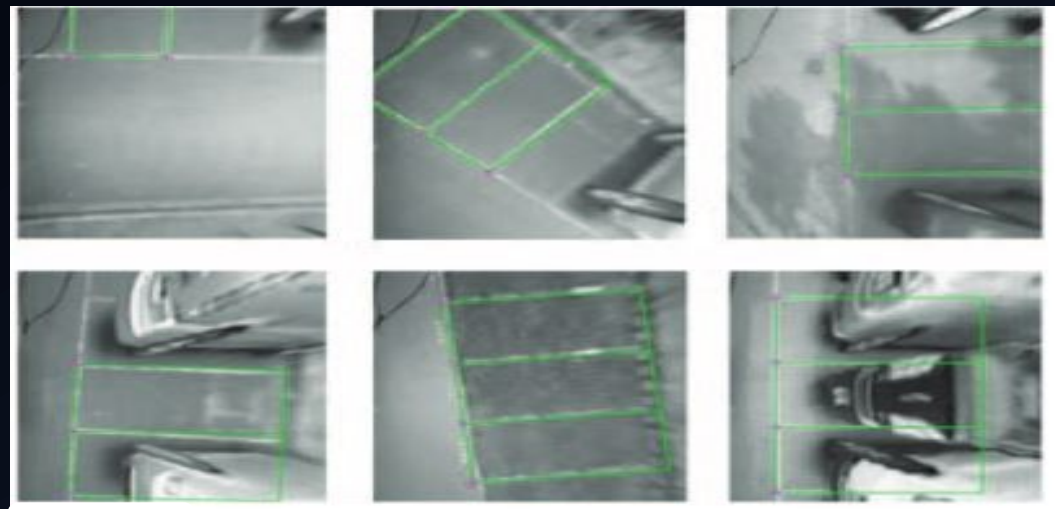


figure5 The proposed method detection experiments of the parking space in different scenes.

Empty Parking Space Extraction.
Figure 6 is the experiment result of empty parking space extraction. In this figure, the red rectangles are occupied parking spaces while the green ones indicate empty parking spaces.

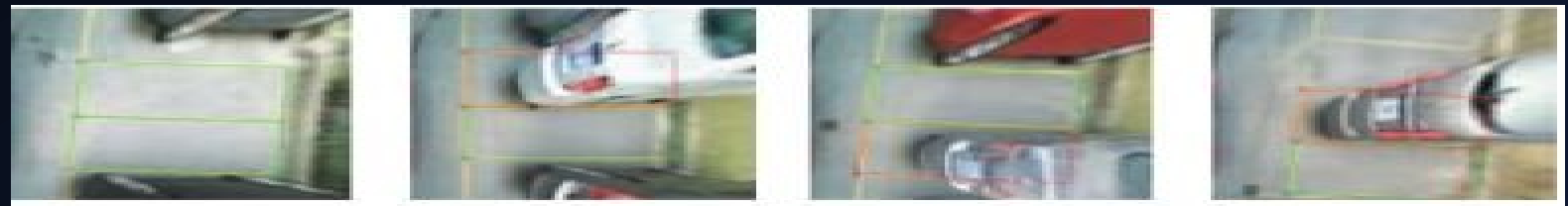
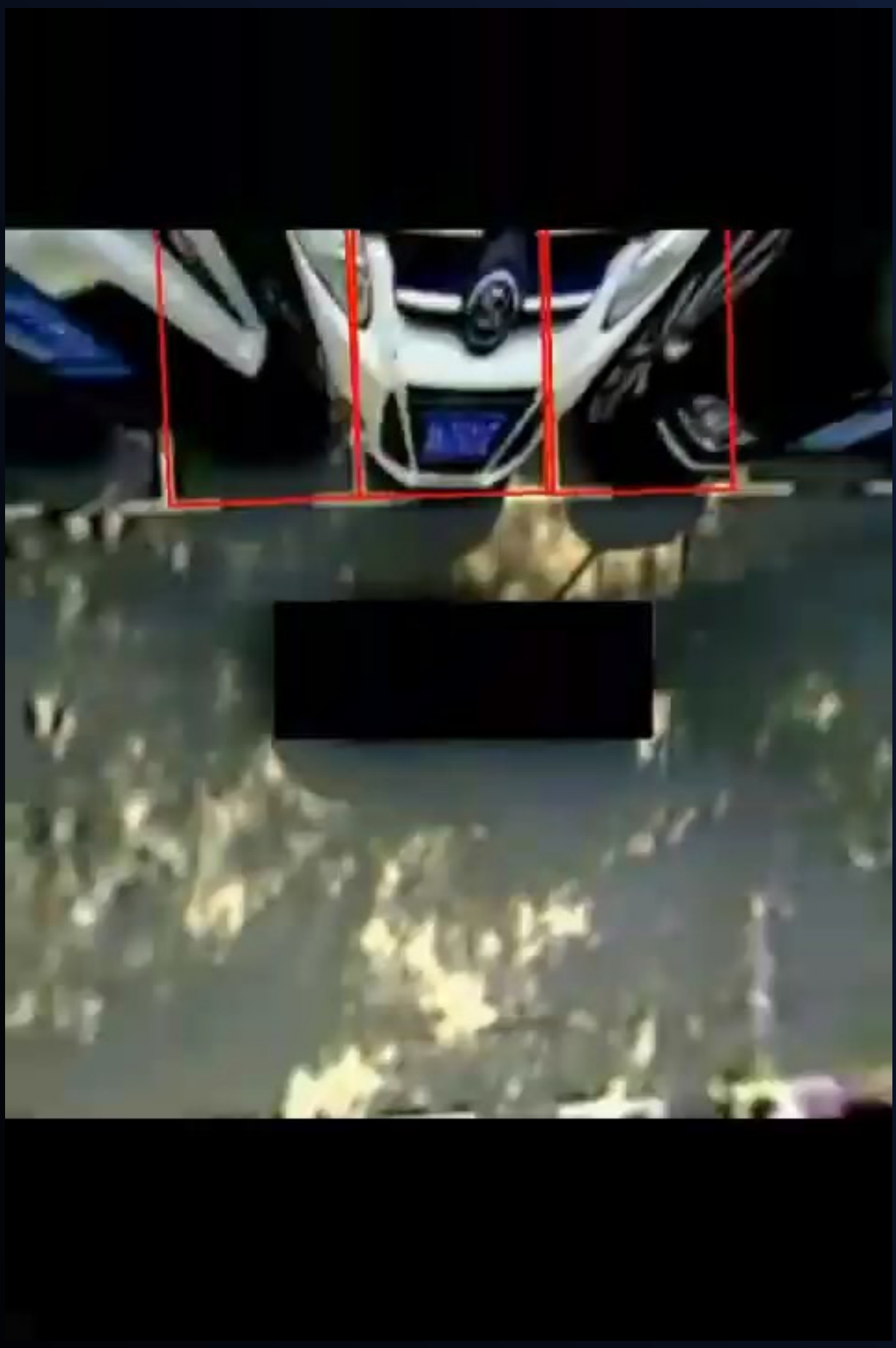


figure6 Experiment result of free parking space.



References:

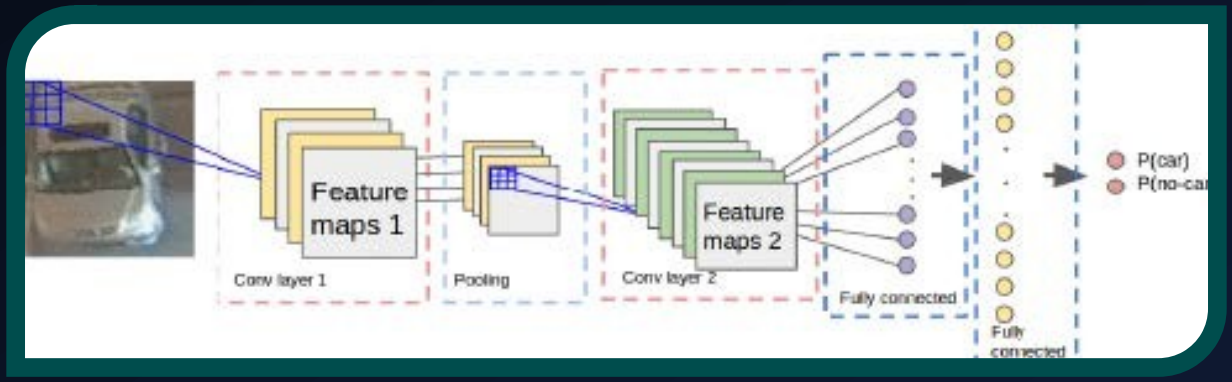
1-Research Article Automatic Parking Based on a Bird’s Eye View Vision System.
2-Context-Based Parking Slot Detection With a Realistic Dataset

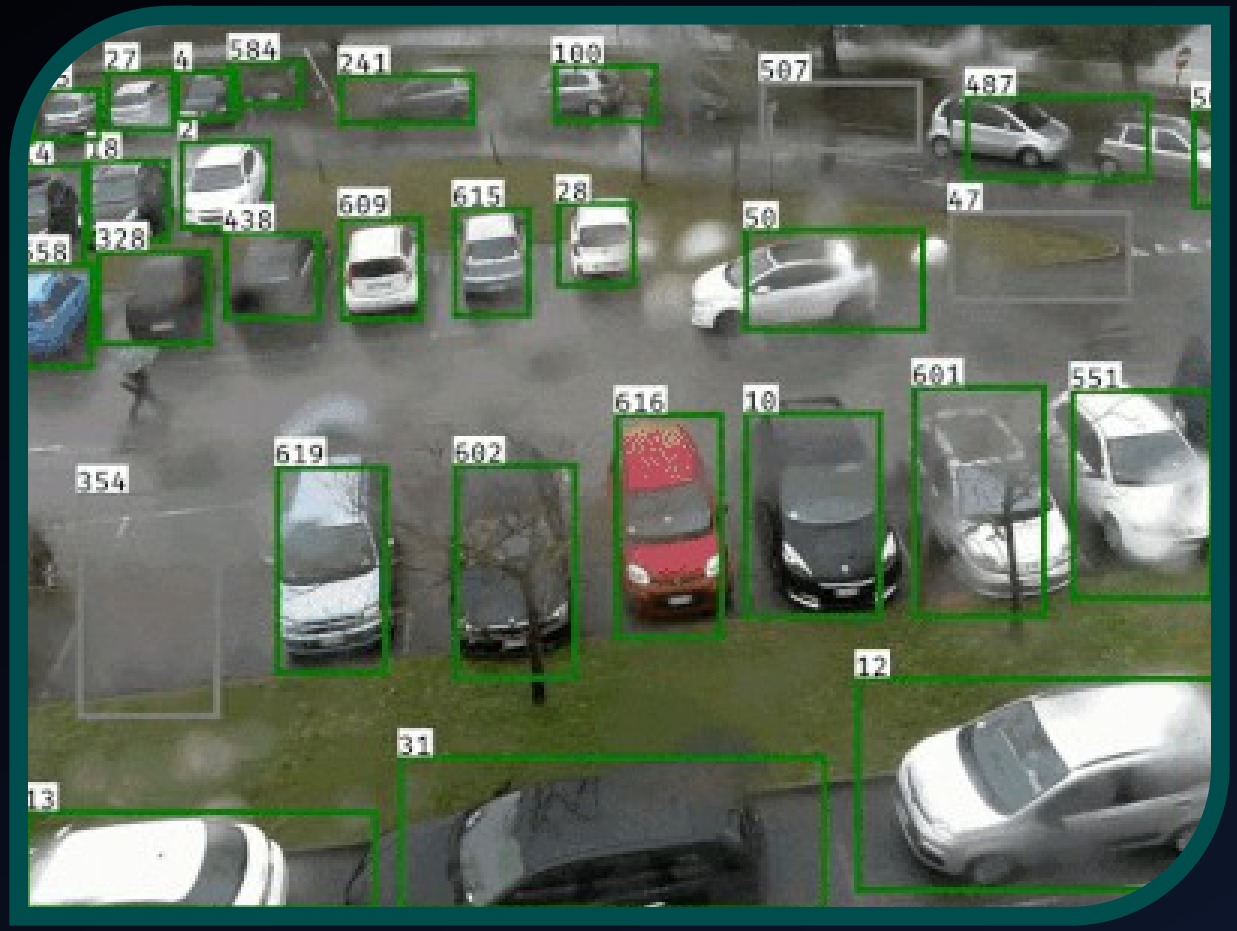
3- Vision-based

- By using vision-based systems than the sensor-based has two advantages: versatility and lower cost. Rather than using one ground sensor per parking space a smart camera can be used which decreases the cost. Compared to the other systems the accuracy of this system is far better than the others. Therefore, rather than using the other systems it is better to go with the vision-based systems which gives high accuracy results.

Advantages	Disadvantage's
Good for car searching in large parking lots.	Has no real intelligence: follow rules.
Easy for lot occupancy detection, parking space recognition, parking charges collection etc.	Software must be carefully selected and implemented.
Easily adaptable to inspect different products	The lighting conditions must be well designed and implemented.
Maintains consistency of inspection.	
Can be reprogrammed off-line on another machine.	

- **CNNs:**
- It used to recognize in real-time the occupancy of the parking space which includes the Quality, accurate, efficient and the scalable answer
- CNN provides a solution for disturbances like partial occlusions, presence of shadows, variation of
- light conditions and exhibits great performance.
- In classification it needs computational resources which makes it appropriate to the embedded environment such as Raspberry Pi.
- A convolutional neural network is a combination of a large number of hidden layers in which each layer does mathematical computations on the image and gives an output image that is given in input to the network.
- the CNN has convolutional layers, which are able to model the spatial correlation of neighboring pixels better than the normal fully connected layers. The classification output is given by the input which is given at the training phase. The model takes lots of training time and also a bit expensive but after the training is done the detection from the network is quite fast and efficient





- **Related Work**
- Author: G. Amato, F. Carrara, F. Falchi, C. Gennaro and C. Vairo
- In this paper, they evaluated the approach for the parking detection, by making using of CNNs to detect the parking lot occupancy by using Raspberry Pi camera.

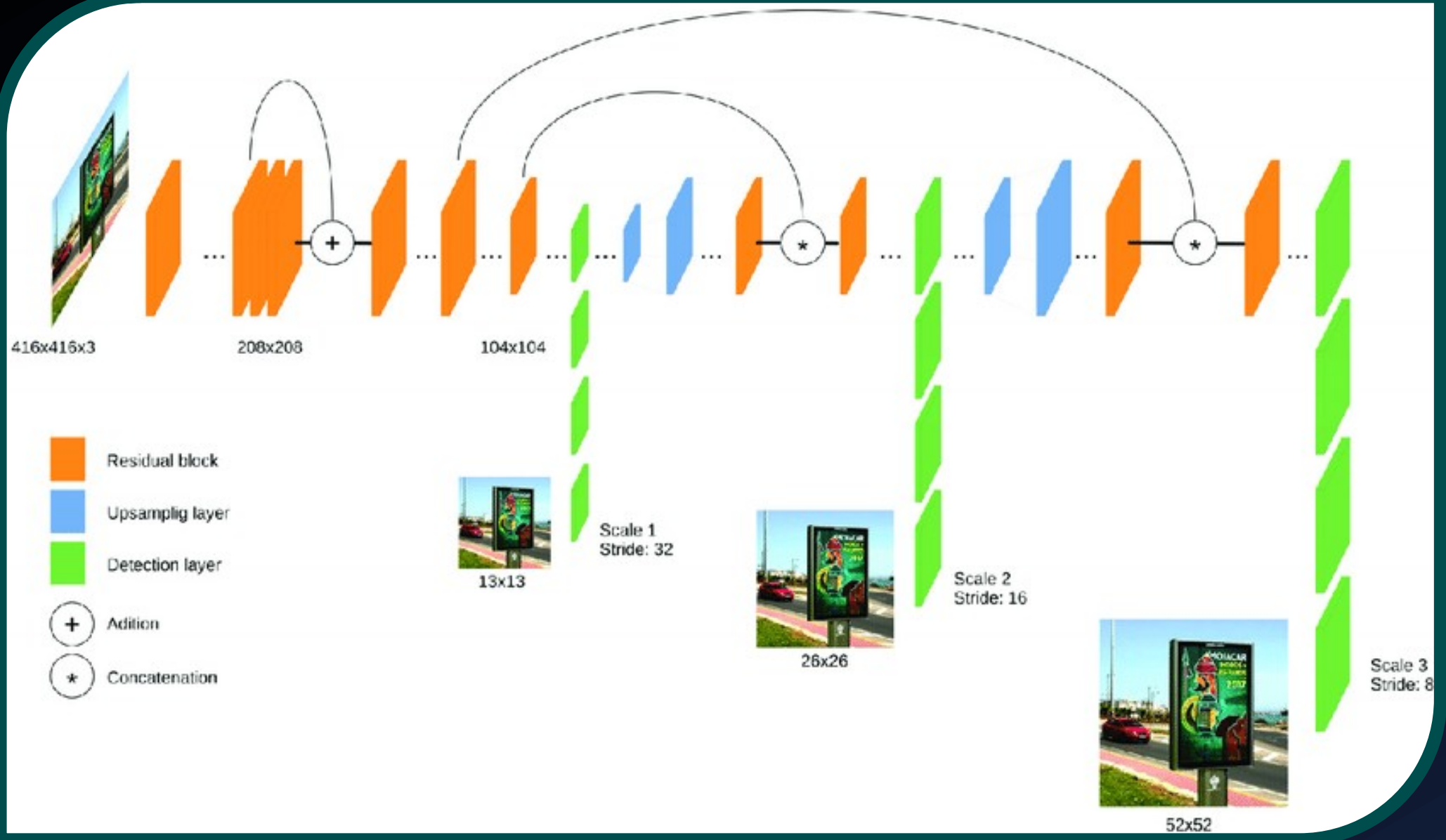
YOLO V3

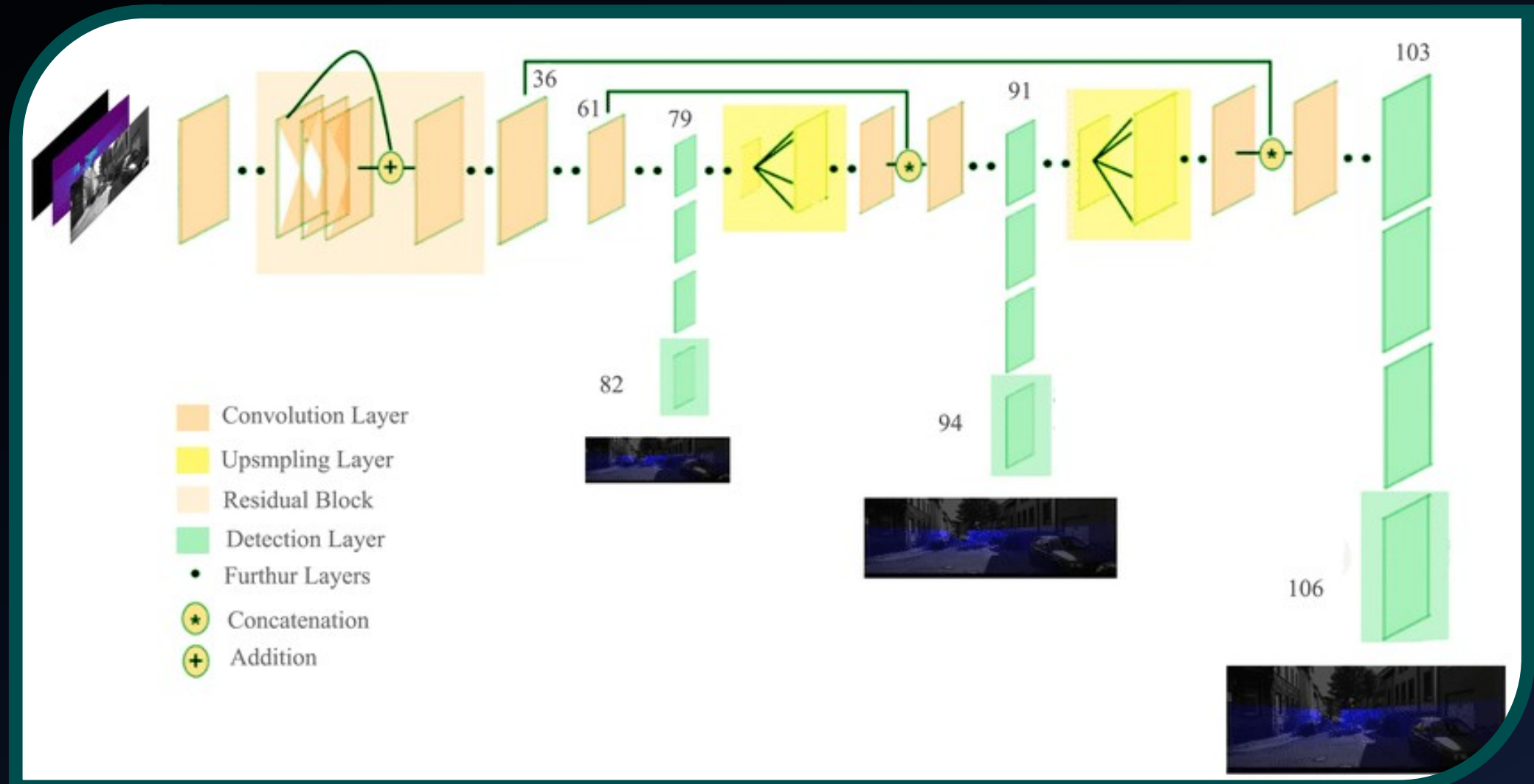
- You Look Only Once, known as Darknet.

- In the past years, YOLO 9000 used to be the fastest which is also one among the most accurate algorithm. But, before a couple of years it is not the best accurate algorithms when compared with the other algorithms like SSD(Single Shot Multibox Detector) and Retina Net which are the best for accuracy
- Then the next version of the YOLO which is YOLOv3 has been released with some of the changes in the internal architecture

YOLO V3

- is one among the faster algorithms for the object recognition.
- YOLO machine learning algorithm is a more accurate version of the original ML algorithm
- YOLO is implemented using the OpenCV deep learning libraries.
- YOLOv3 architecture consists of “53” layered network which is originally trained on ImageNet. Then mainly for the part of detection, “53” more layers are added onto it, and is given by a “106” fully convolutional underlying architecture for YOLOv3
- In the YOLOv3 algorithm the detections can be done at 3 different scales which is considered as the most important feature in the entire algorithm.
- YOLOv3 architecture the detection of the object in the image is done at 3 different sizes and at 3 different layers in the architecture.


416x416x3
208x208
104x104
Residual block
Upsamplig layer
Detection layer
Addition
Concatenation
13x13
Scale 1
Stride: 32
26x26
Scale 2
Stride: 16
52x52
Scale 3
Stride: 8


36
61
79
82
91
94
103
106
Convolution Layer
Upsmpling Layer
Residual Block
Detection Layer
Furthur Layers
Concatenation
Addition

YOLO V3

- It is fully convolutional neural network which gives the output by applying a “1 x 1” kernel on a feature map .
- The shape of the detection kernel is given by “1 x 1 x (B x (5+C))”.
- B represents the no. of the bounding boxes which are present in a cell on the feature map which can predict.
- 5 represents the 4 boundary box attributes and 1 for the confidence of the object
- C denotes the no. of classes
- The YOLOv3 which is trained on COCO dataset
- It gives B = 3 and C = 80, so which gives the size of the kernel as

“1 x 1 x (3 x(5+80))” –> “1 x 1 x 255”.

Example

- the image which is given as input is of size “416 x 416”,
- as mentioned earlier the algorithm makes the predictions at 3 different layers of the network which are done by down sampling the data by 32, 16 and 8.
- In the YOLOv3 algorithm detection is done at 3 different layers.
- 1 st detection will take place at the 82th layer, by giving a feature map of “13 x 13 x 255”. Stride 32
- 13 x 13 → 416/32=13

- 2nd detection will take place at the 94th layer, by giving a feature map of “26 x 26 x 255”. Stride 16
- 26 x 26 → 416/16=26

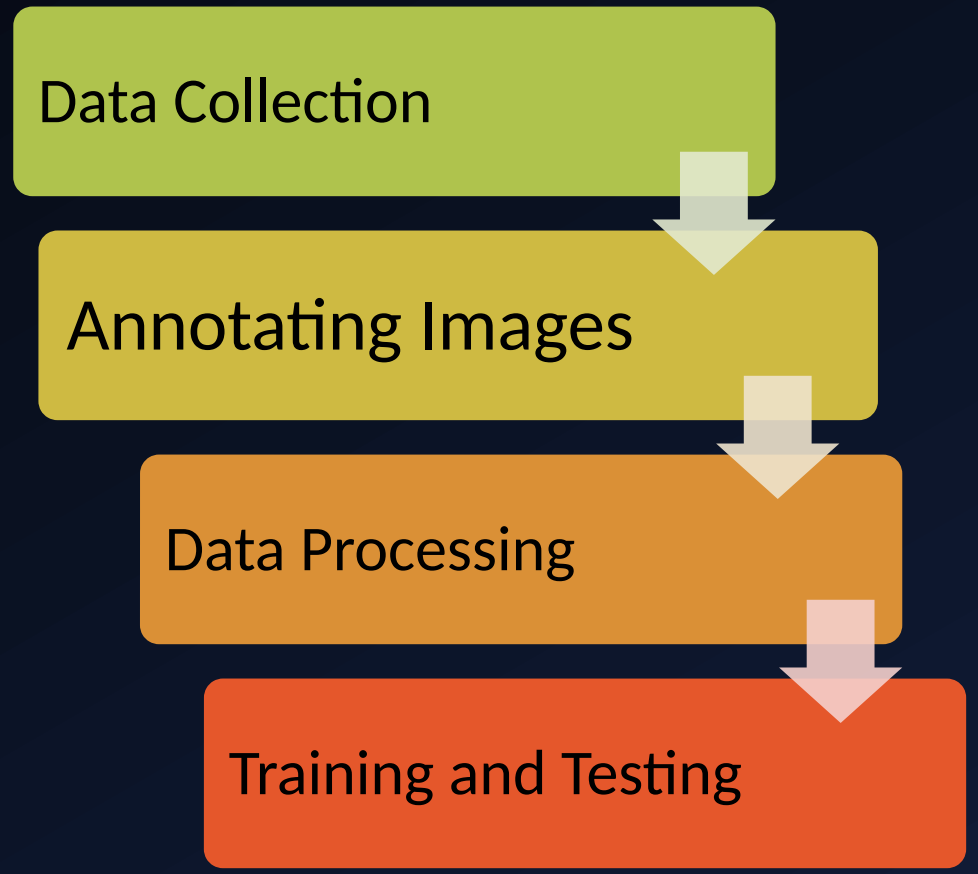
- 3rd detection will take place at the 106th layer, by giving a feature map of “52 x 52 x 255”. Stride 8
- 52 x 52 → 416/8=52

- . YOLOv3 makes use of 3 anchor boxes,
- . YOLOv3 totally makes use of 9 anchor boxes,
- These anchor boxes are arranged one by one it calculate the probability of the object
- YOLOv3 makes predictions 10times the number of boxes done by YOLOv2.
- Each bounding box in a YOLO algorithm consist of four descriptions:
- Centre of the bounding box ($b_{x,}$ b_{y})
- Width (b_{w})
- Height (b_{h})
- Class of object (c)
- The Probability that there is an object in the bounding box (p_{c})

Methodology

Hint

- To training and finding a car in the parking space is done using convolutional neural networks. In the proposed method, this would like to collect the images of cars parked in the outdoor parking lot and create a dataset with all the images so that the neural network can detect the car present in the parking space or not accurately
- We Have 4 Steps To Execution Image:



Data Collection

- The PKLot dataset contains 12,416 images of parking lots and 695,899 images of parking spaces segmented from them, extracted from surveillance camera frames. There are images on sunny, cloudy, and rainy days and the parking spaces are labeled as occupied or empty. We have converted the original annotations to a variety of standard object detection formats by enclosing a bounding box around the original dataset's rotated rectangle annotations.
- All images were acquired at the parking lots of the Federal University of Parana (UFPR) and the Pontical Catholic University of Parana (PUCPR), both located in Curitiba, Brazil. The protocol used to construct the PKLot dataset is composed of three steps, as follows:
- 1- Image Acquisition
- 2-Labeling
- 3-Segmentation

Picture 1 Park 1
Data Collection

Picture 2 Park 1
Data Collection

Picture 3 Park 1
Data Collection

Picture 4 Park 1
Data Collection

Picture 1 Park 2
Data Collection

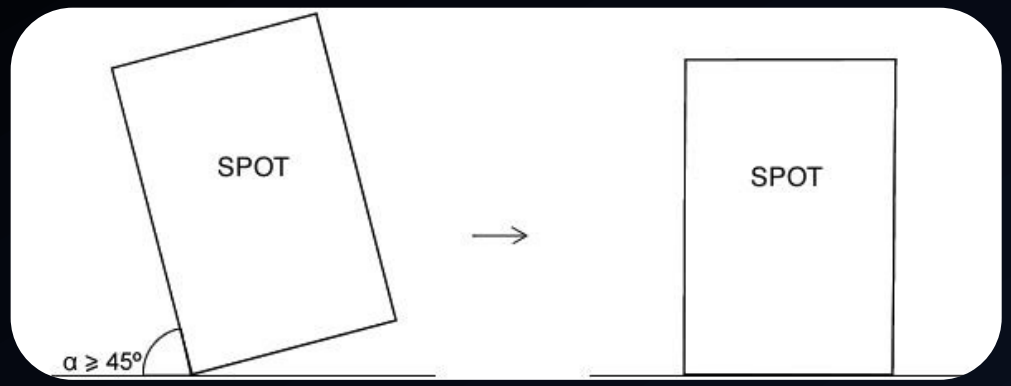
Picture 2 Park 2
Data Collection

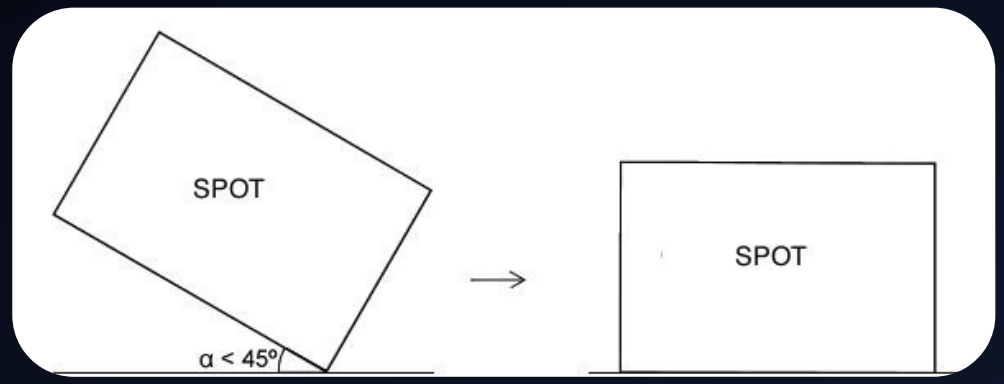
Picture 2 Park 2
Data Collection

Picture 5 Park 2
in another View
Data Collection

Picture 4 Park 2
Data Collection

the slope of the rectangle containing the parking space image was modied according to the scheme presented in Figure





References:

[Almeida et al.(2013)Almeida, Oliveira, Silva, Britto Jr & Koerich] Almeida, P., Oliveira, L. S., Silva, E., Britto Jr, A. S., & Koerich, A. (2013). Parking space detection using textural descriptors. In Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on (pp. 36033608). doi:10.1109/SMC.2013.614.

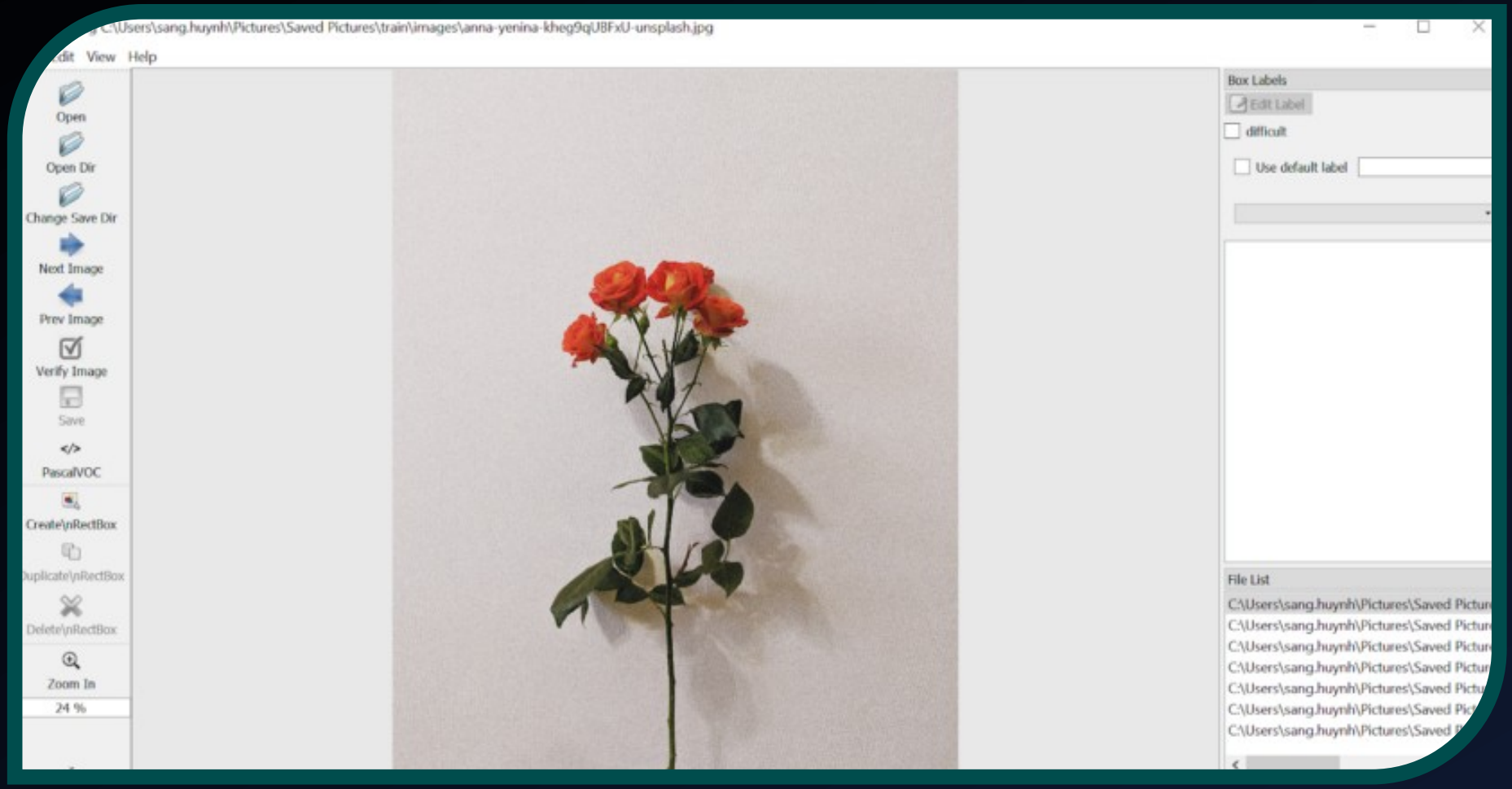
Annotating Images

- We annotated images with two different classes which are “Empty” and “Occupied”.
- We Store Data In a Label:
- For labelling the image we used **LabelImg** software which is an open source.
- In this software the images can be annotated easily as it is user friendly.
- Firstly, the software contains some pre-defined classes which should be changed according to our classes, we have named the classes as “Empty” and “Occupied”.
- The software is set to YOLO format and we annotated the images where there is no car in a particular slot as “Empty” and where there is a car as “Occupied”.
- The data of the bounding boxes that we annotated along with the two classes are saved in the txt format which will save the coordinates of the bounding boxes.

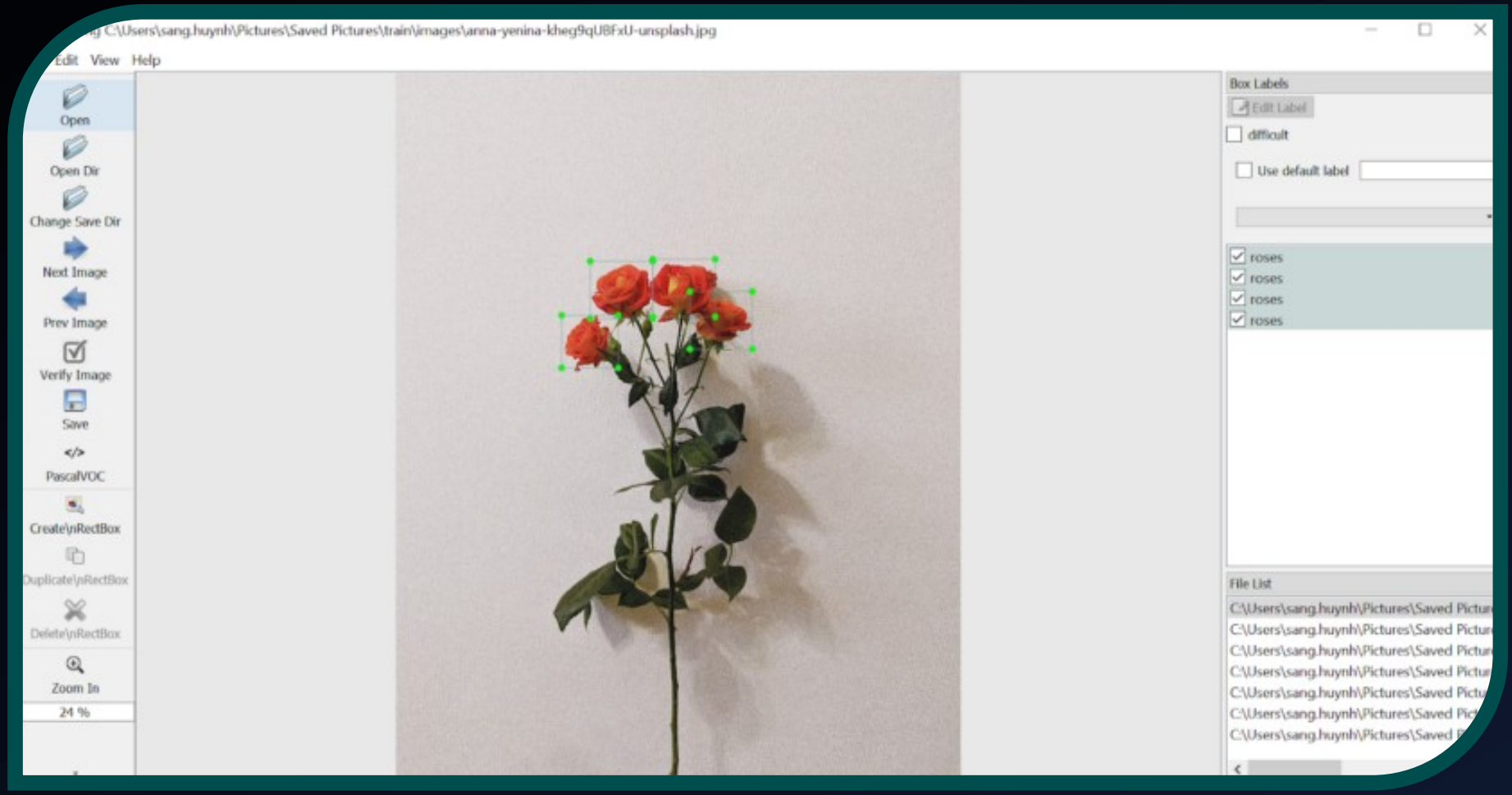
LableImg Tool

- **What is Data labeling?**
- One of the most important part in the data science process is data pre-processing which makes data scientists put lots of time and effort into. Particularly, in the context of machine learning, data pre-processing requires a step of labeling.
- This is the step to detect and label data sample into multi-classifications so that the labeled data can be used in further machine learning process.
- Two typical labeling data type in ML projects are text and image annotation.

- **What is LabelImg?**
- **LabelImg** is a graphical image annotation tool which provides images with bounding boxes after labeling. It is written in Python and uses Qt (one of the most common GUI for python) for its graphical interface.
- The output labeled data are saved as XML files in PASCAL VOC format, the format used by ImageNet. Also, YOLO format is supported in this tool.



the LabelImg User Interfaces when opening



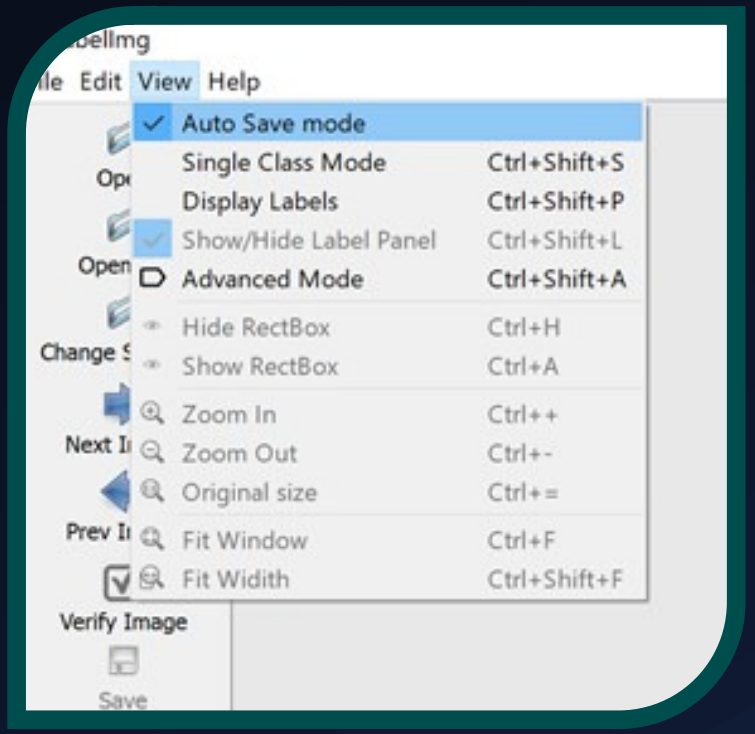
the LabelImg User Interfaces when opening

- **Why do we use LabelImg?**
- It is open-source then it is free to use.
- It has both online and offline versions. The offline interface is user-friendly and allows the user to make fast annotations.
- The output annotation images are saved separately as XML files along with Pascal VOC format or YOLO format. These are common formats supported by many ML packages.
- After installation, it is easy to use with simple instructions.
- It can process a large number of images when compared to other free apps such as **Cloud Vision API** (the free-offline version is limited up to only 2000 image files) or **Labelbox** (the free version only allows maximum of 5000 images to be processed).
- It can be run in multiple operating systems such as Linux, Mac or Windows.
- It is written in Python then it can interact with other python packages and is easy to be used in python applications for further development.

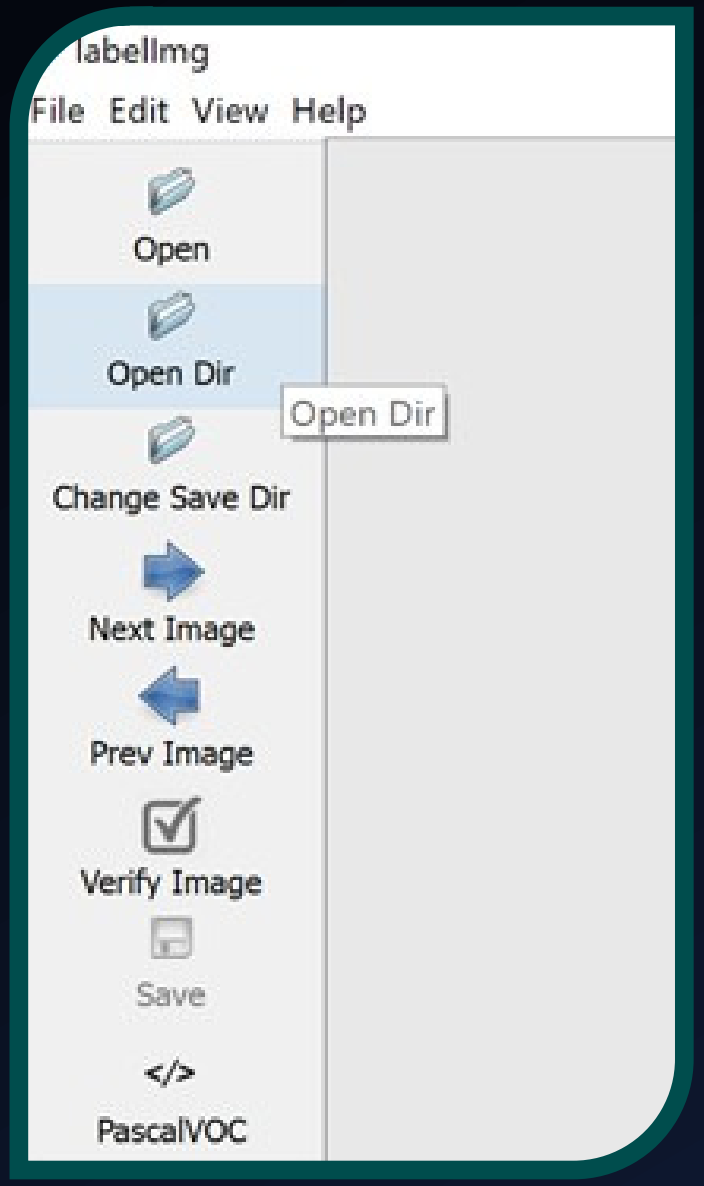
- **this tool still has some limitations:**
- Bounding box (rectangle shape) is the only region shape to be used. Therefore, it cannot give an exactly detection for complicated objects with curved lines…
- The installation requires some tricks we need to know. Therefore, I will present how to install this tool step by step in the next session.

- **Installation**
- **Prerequisites:**
- The installation below is used for **Windows OS** with **Anaconda**. Therefore, we need to install **Anaconda** to have an **Anaconda terminal**.
- Follow the link below to install **Anaconda** in Windows:
- https://docs.anaconda.com/anaconda/install/windows/
- **Installing LabelImg:**
- If We are using Python 3, the easiest way to install LabelImg is using pip. Simply open your command line and run: ‘pip3 install labelImg’. To launch, run ‘labelImg’ in your command line prompt

- **Organise your dataset**
- Once you have opened LabelImg:
- Create a folder titled “images” and copy all of your images there
- Create another folder titled “annotations”
- Go to the LabelImg “View” menu and check that “Auto Save Mode” is selected



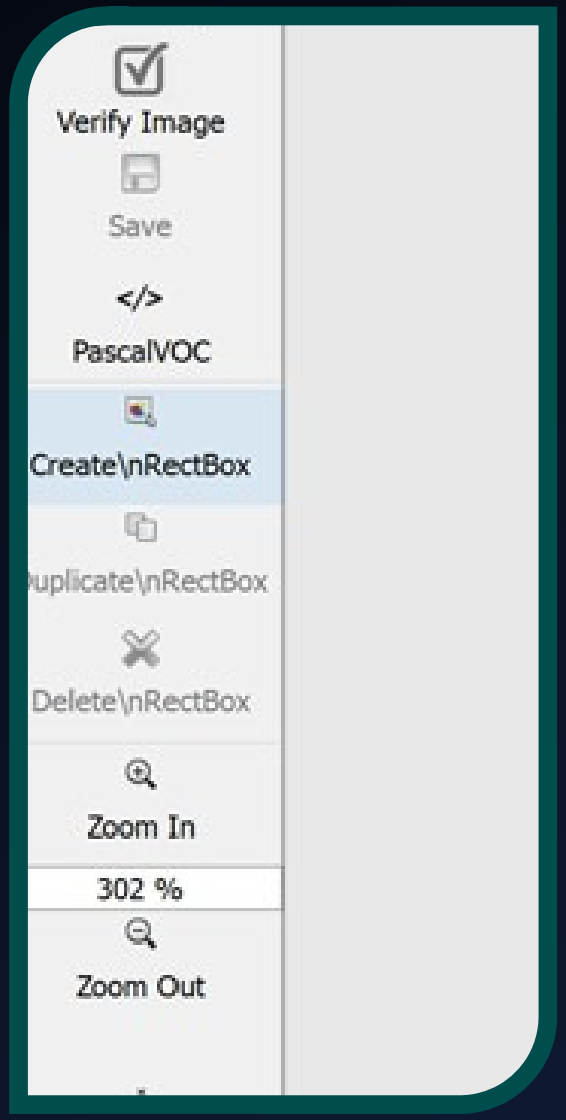
- Click on “Open Dir” on the left hand side and select your “images” directory



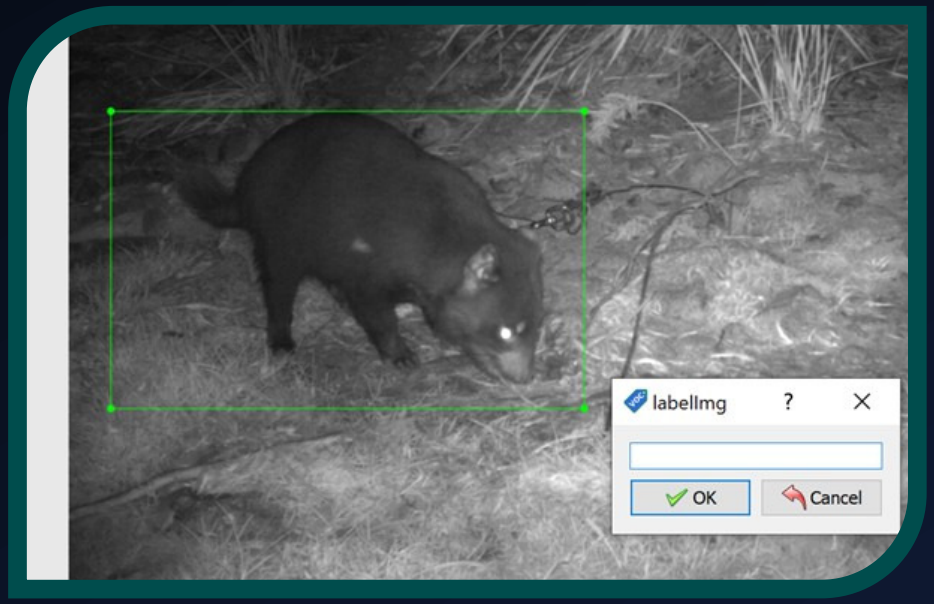
- LabelImg will then ask where to save annotations to, select your “annotations” directory.
- If no image comes up click the “Next Image” button a few times and it should begin to flick through your images.

- Before labelling your images, check which label file format you would like to use. VOC XML is a more of a universal standard when it comes to object detection. LabelImg also supports YOLO format if you would prefer to use that. To change just click on the “</> PascalVOC” or “YOLO” box on the left hand side.

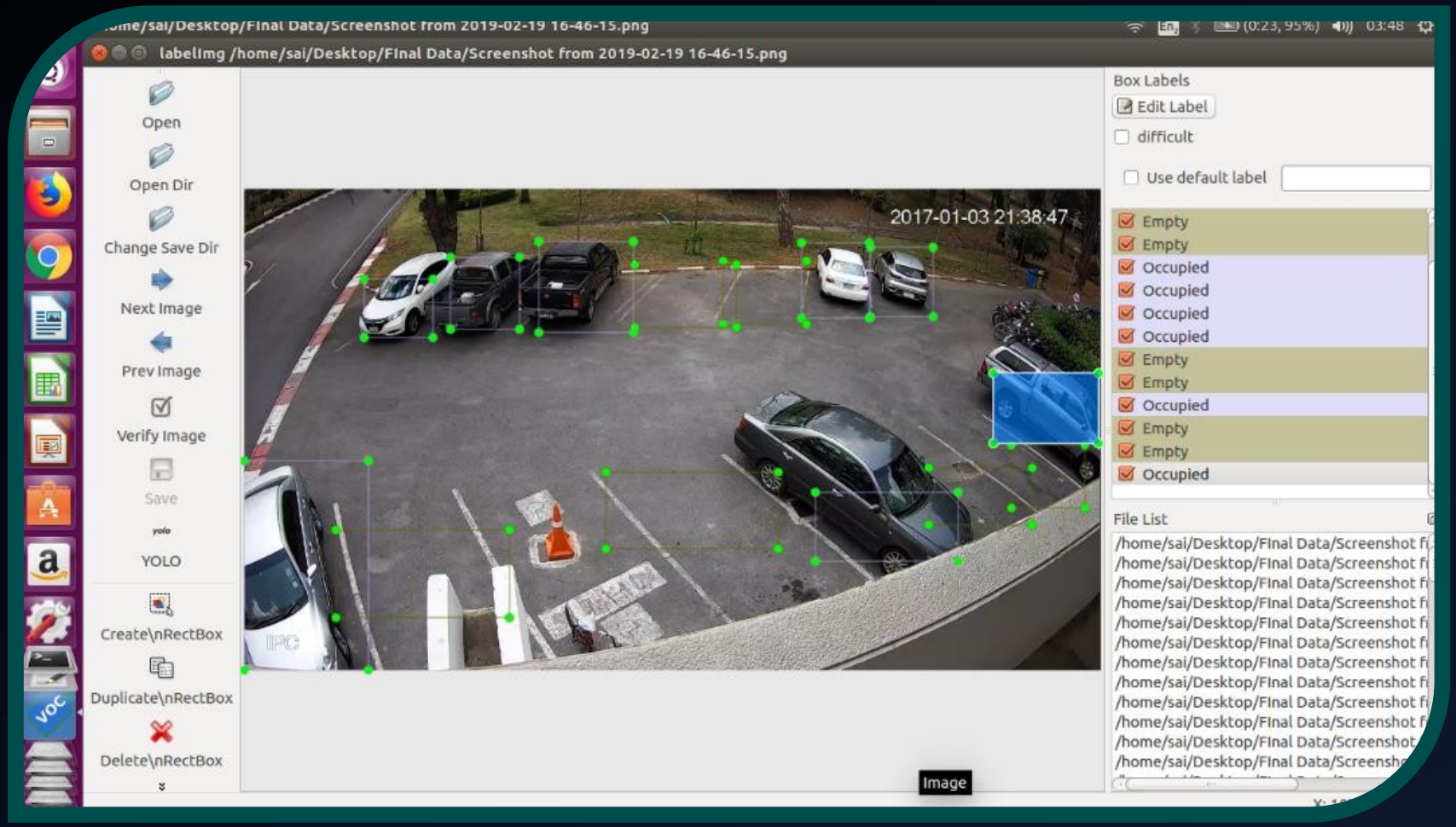
- **Annotating your Dataset**
- Click on the “Create n\RectBox”



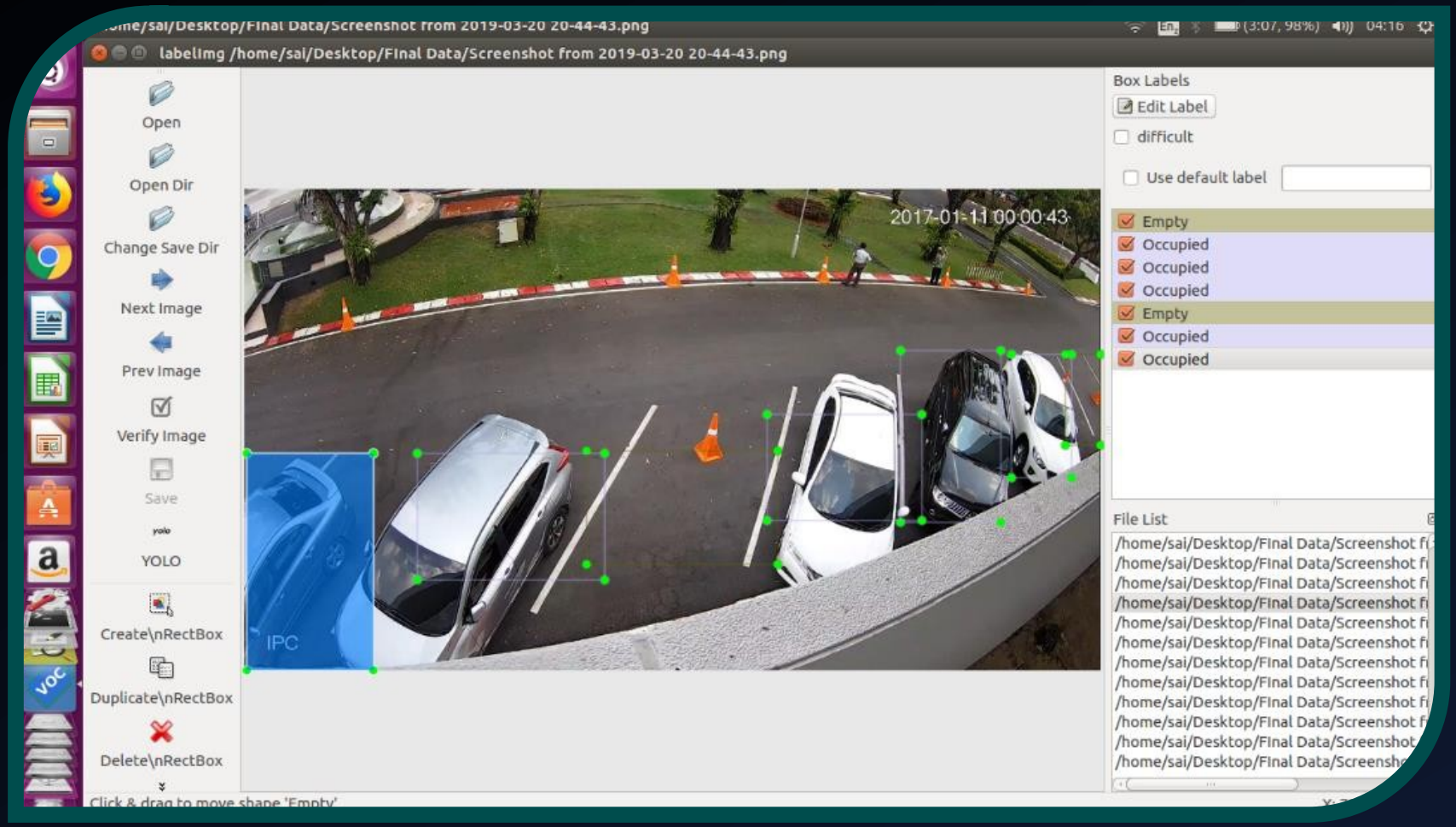
- Use your mouse to draw a box around the object you are wanting to label
- Enter the name of the object in the pop-up box and select “OK”
- If there is more than one object in the picture select “Create \nRectBox” again and annotate all the objects
- Click on the “Next Image” button and repeat the process until you are finished annotating all your images. All the image annotations will be automatically saved in your “annotations” folder. They will be xml files with names corresponding to your image names.



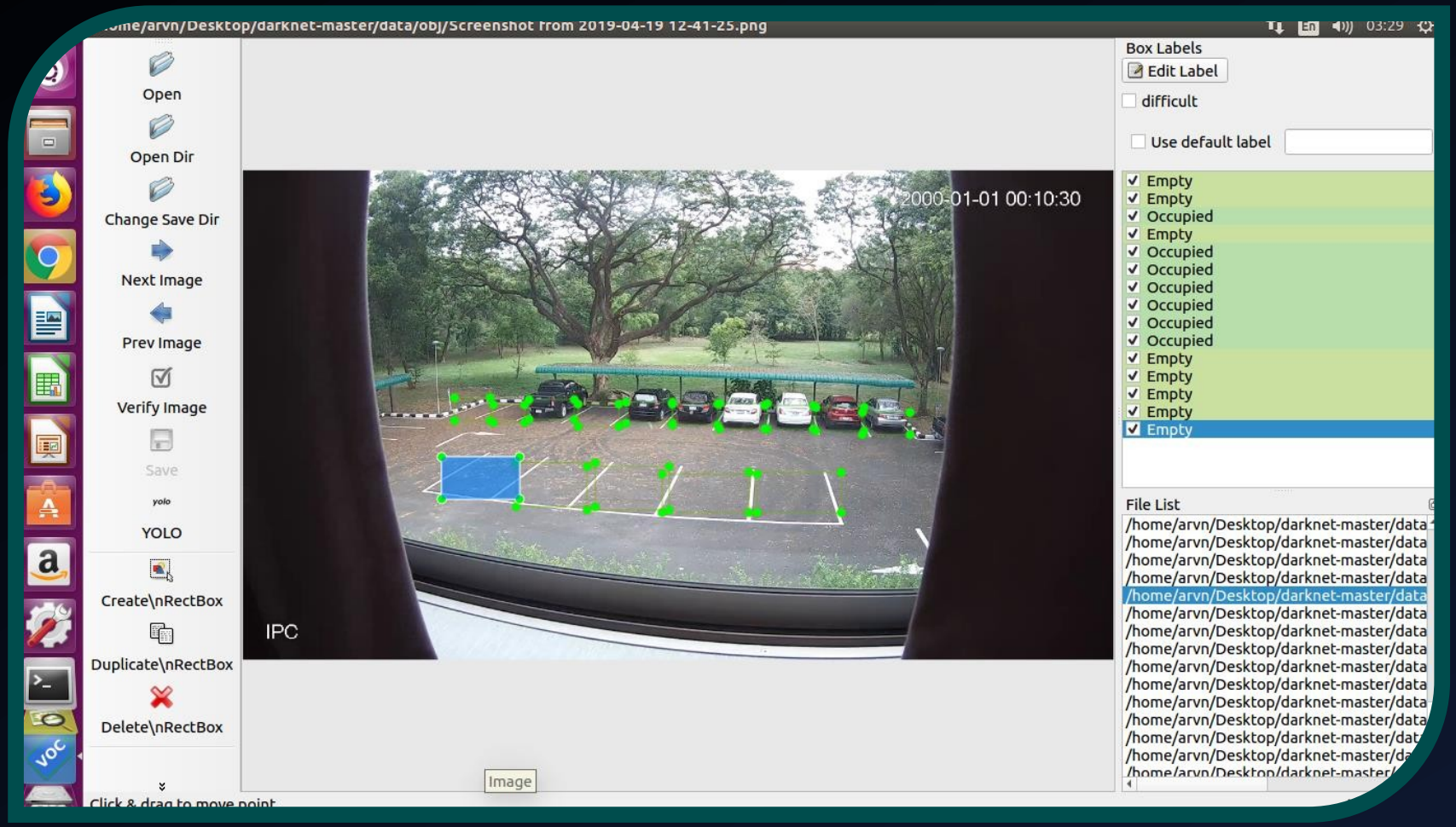
Examples Image Labelled using LableImg Tool



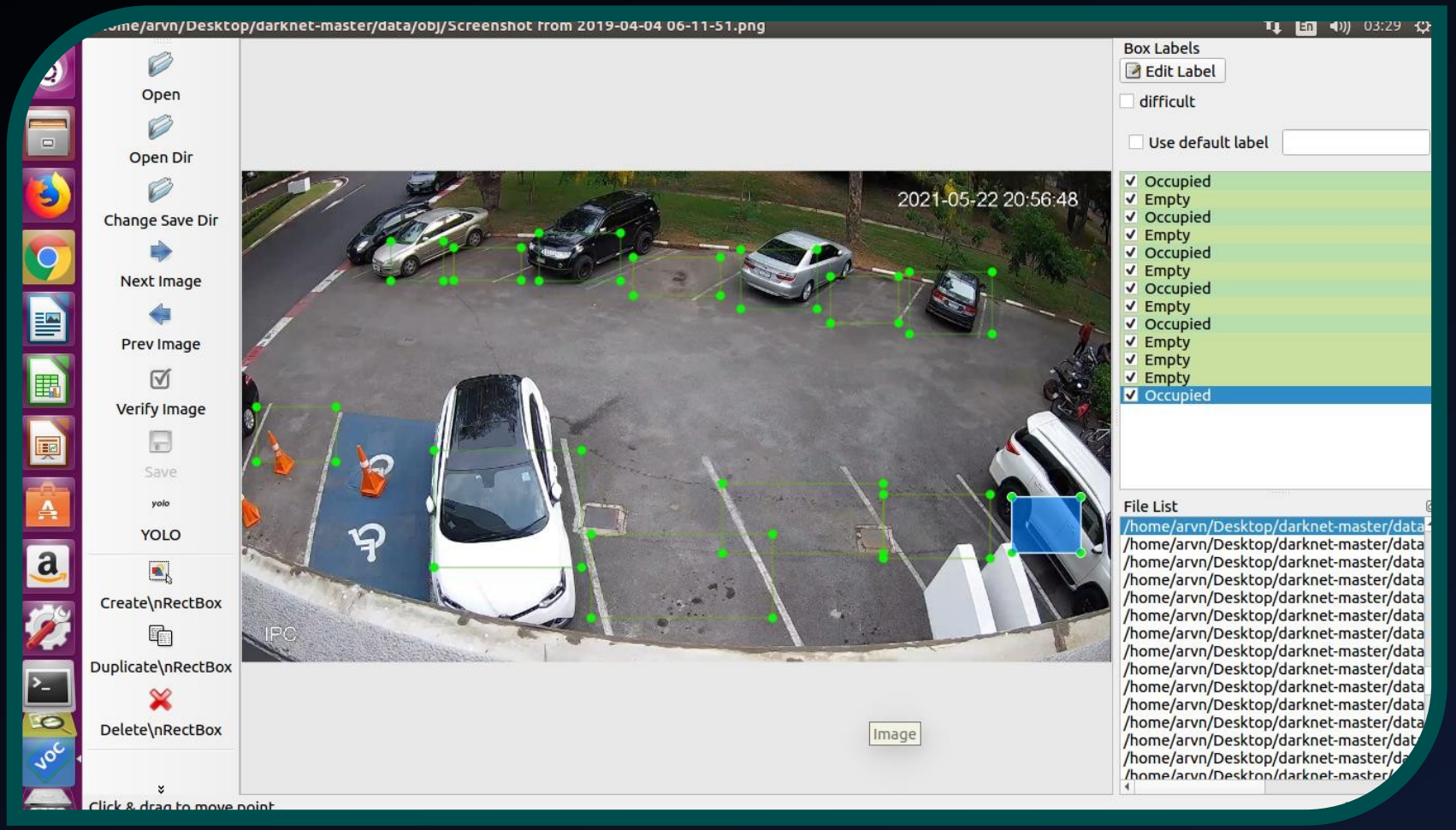
Picture 1 : Image labelled using LabelImg Tool



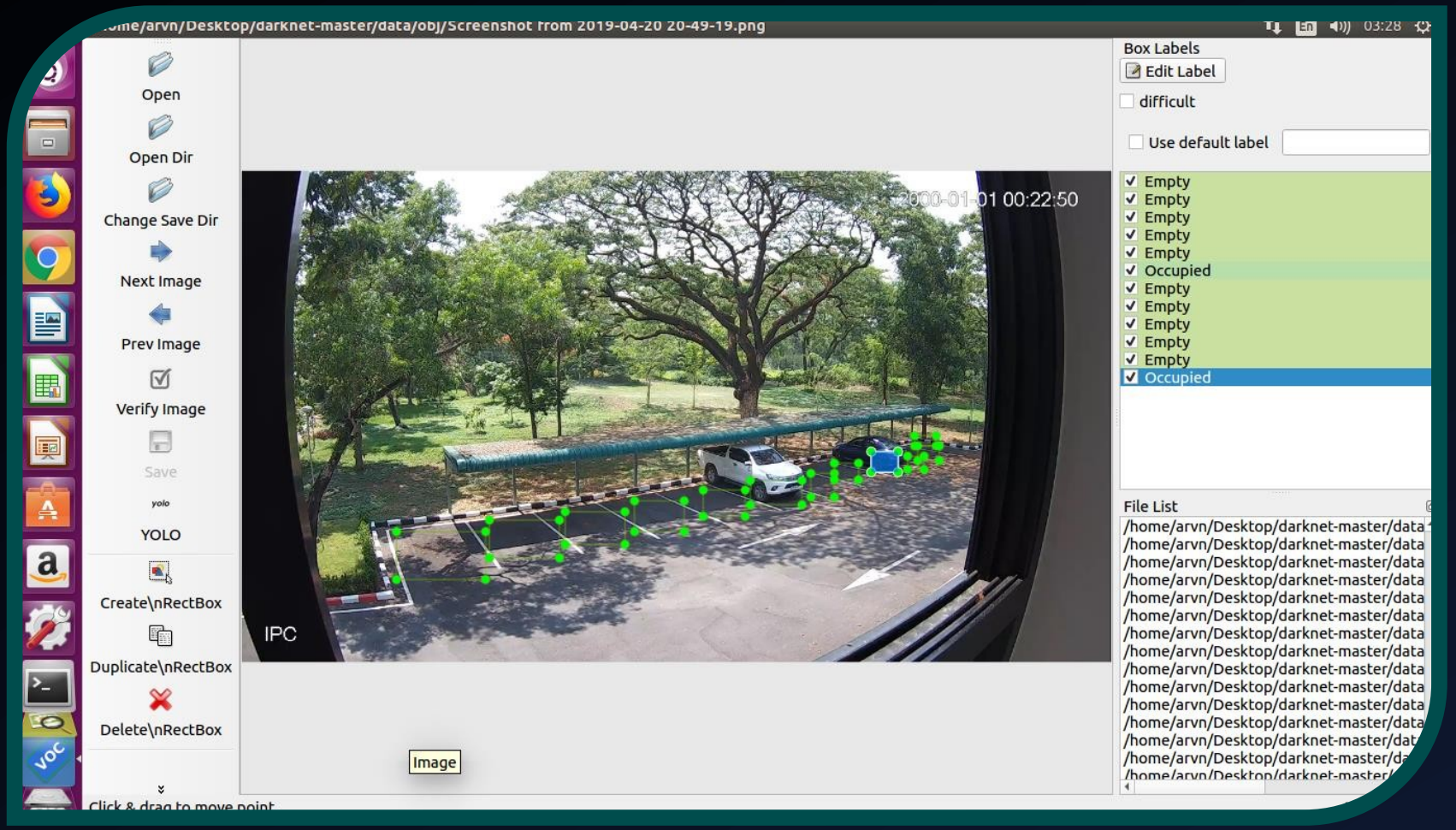
Picture 2 : Image labelled using LabelImg Tool



Picture 3 : Image labelled using LabelImg Tool



Picture 4 : Image labelled using LabelImg Tool



Picture 4 : Image labelled using LabelImg Tool

Data Processing

- For YOLOV3 detector, there is a different format for the values to be fed into the model. These values are saved in the txt format which contains some of the specified classes and some values. These values can be as below.
- The order of the values in txt file is class, x, y, w, h.
- “where x = Absolute x / width of total image.
- y = Absolute y / height of total image.
- w = Absolute width / width of the total image.
- h = Absolute height / height of total image.”
- Where Absolute x, Absolute y, Absolute width, Absolute height are given below
- “Absolute x = (X_{min} + (Absolute width/2))
- Absolute y = (Y_{min} + (Absolute height/2))
- Absolute width = abs (X_{max} - X_{min})
- Absolute height = abs (Y_{max} - Y_{min})”

```
1 0.767936 0.193359 0.080527 0.134115
0 0.692167 0.209635 0.085652 0.106771
1 0.613104 0.188151 0.082723 0.175781
1 0.516837 0.196615 0.117130 0.192708
1 0.395681 0.198568 0.114934 0.201823
1 0.280015 0.213542 0.085652 0.182292
1 0.173865 0.245443 0.092972 0.134115
1 0.069180 0.776693 0.132504 0.446615
1 0.204978 0.772786 0.206442 0.160156
0 0.525622 0.655599 0.224012 0.165365
0 0.747804 0.715495 0.169107 0.128906
0 0.778551 0.563802 0.135432 0.132812
0 0.925695 0.602865 0.106149 0.101562
1 0.932284 0.438802 0.128111 0.161458
```

```
0 0.979868 0.456380 0.040264 0.222656
1 0.926428 0.477214 0.063690 0.279948
1 0.830893 0.524089 0.118594 0.345052
1 0.698389 0.574870 0.146413 0.371094
0 0.517204 0.621094 0.218887 0.226562
1 0.309663 0.691406 0.247438 0.270833
1 0.074305 0.759766 0.145681 0.451823
```

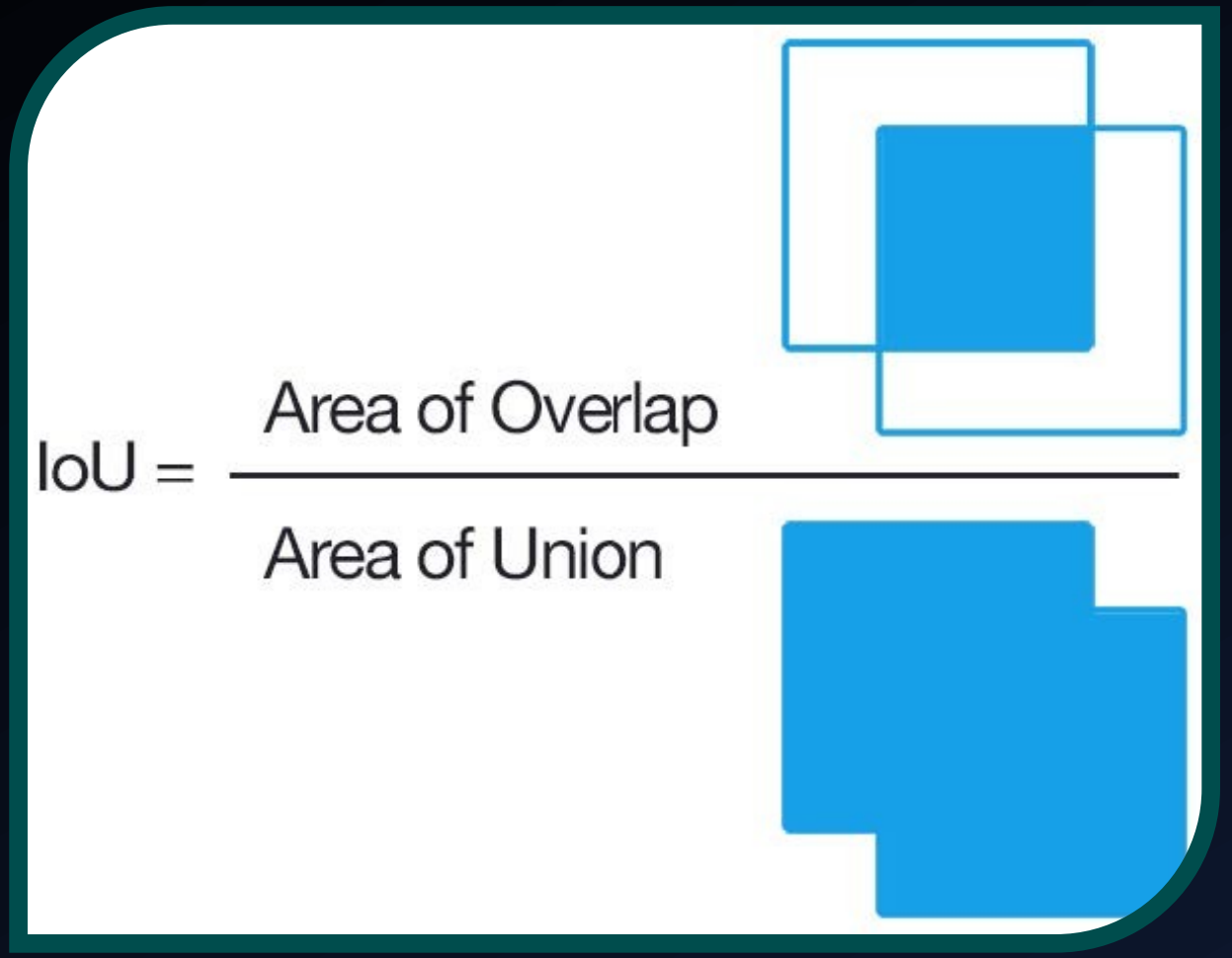
YOLO format of txt files

Note: the number ‘0’ denotes the class “Empty” and the number ‘1’ denotes the class “Occupied” and the other values denotes the metrics x,y,w,h.

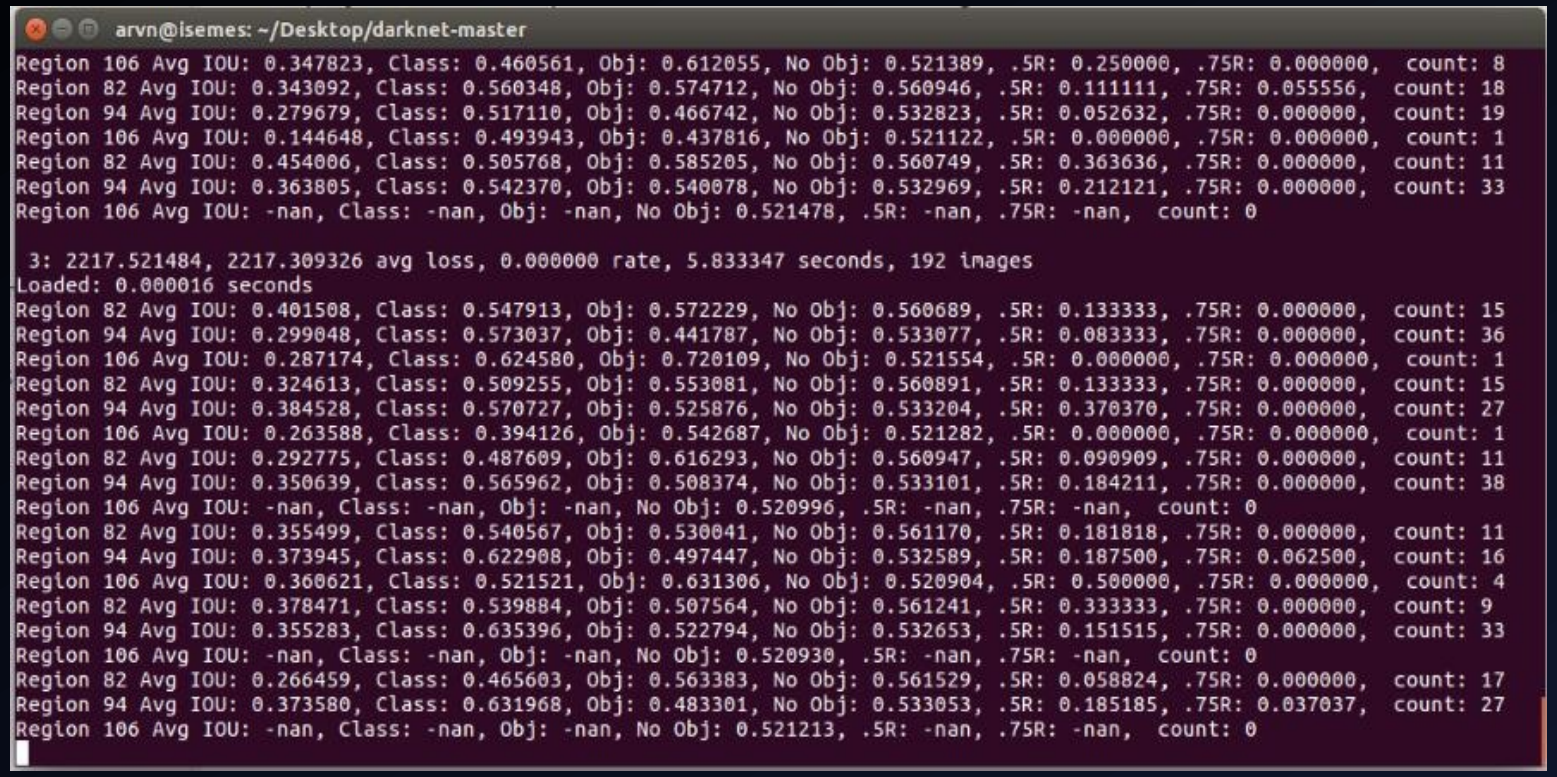
Training and Testing

- As the deep learning requires hardware with high end specifications for training.
- We have used the pre-trained weights of YOLOV3 which are trained on COCO dataset which consists of 80 classes and images are more than 80000. The training of our model on the dataset was done by the previous steps. The filters of YOLOV3 configuration file should be set according to the below YOLOV3 filters formula
- • “Filters = (classes + 5) * 3”
- As there are two classes in our case, the classes are set to our number of objects that is two in each of 3 yolo layers, the filters which are in convolutional layer just before the 3 yolo layers are set to 21. The YOLOv3 configuration file consists of batch number which is set to 64, subdivisions to 16, learning rate to 0.001, saturation to 1.5 and exposure to 1.5. The resolution of the input image is resized to a width and height of 608*608. And the training of our model is done, but as YOLOv3 does not limit the number of iterations to be done, the system continuously trains the model. Usually 2000 iterations are sufficient for each class(object), but the iteration number should not be less than 4000 in total. But, for a better precision the training should be stopped for the iteration where the average loss no longer decreases or remains constant. So, when the training should be actually stopped depends upon the average loss function.

- The average loss function changes gradually from a higher value to the lower value and remains constant, then the training can be stopped.
- The weights file is generated and saved for every 1000 iterations and this can be changed in the configuration file for our desired number of iterations.
- After the training is done the mAP (Mean Average Precision) is calculated for the validated images.
- This should be done for every weight so that we can know the mAP for each and every weight and the weights file whose mAP is higher can be used for testing the image for good detection.
- Thresh is considered as the minimum threshold of the IOU(Intersection Over Union) during the training. IOU(Intersection Over Union) is determined to be the fraction of area of union between the anchor box and the ground truth.



Intersection Over Union



Screenshot taken during training the YOLOv3 model